International procedure specification for Logistics Support Analysis (LSA)

S3000L-B6865-03000-00

Issue No. 2.1

S-Series IPS specifications
Block release 2021

Publishers:

AeroSpace, Security and Defence Industries Association of Europe

Aerospace Industries Association of America (AIA)

Copyright, user agreement and special usage rights

1 Copyright

All correspondence and queries should be directed to:

ASD
Rue du Trône
1050 Brussels
Belgium

2 Agreement for use of the specification S3000L™ suite of information

2.1 Definitions

The **S3000L™ suite of information** means, but is not limited to:

- the International procedure specification for Logistics Support Analysis LSA - S3000L
- S3000L XML schemas
- S3000L UML model
- examples (eg, XML instances, PDF files)
- the input data specification S3000X
- any other software or information on the page titled " Downloads" on the website www.s3000l.org

Copyright holder means AeroSpace, Security and Defense Industries Association of Europe (ASD).

2.2 Notice to user

By using all or any portion of **S3000L™ suite of information** you accept the terms and conditions of this user agreement.

This user agreement is enforceable against you and any legal entity that has obtained **S3000L™ suite of information** or any portion thereof and on whose behalf it is used.

2.3 License to use

As long as you comply with the terms of this user agreement, the copyright holders grant to you a non-exclusive license to use **S3000L™ suite of information.**

2.4 Intellectual property rights

S3000L™ suite of information is the intellectual property of and is owned by the copyright holder. Except as expressly stated herein, this user agreement does not grant you any intellectual property right in the **S3000L™ suite of information** and all rights not expressly granted are reserved by the copyright holder.

2.5 No modifications

You must not modify, adapt or translate, in whole or in part, **S3000L™ suite of information**. You may however add business rules.

2.6 No warranty

S3000L™ suite of information is being delivered to you "as is". The copyright holder does not warrant the performance or result you may obtain by using **S3000L™ suite of information**. The copyright holder makes no warranties, representations or indemnities, express or implied, whether by statute, common law, custom, usage or otherwise as to any matter including without limitation merchantability, integration, satisfactory quality, fitness for any particular purpose, or non-infringement of third parties rights.

2.7 Limitation of liability

In no event will the copyright holder be liable to you for any damages, claims or costs whatsoever or any consequential, indirect or incidental damages, or any lost profits or lost savings or for any claim by a third party, even if the copyright holder has been advised of the possibility of such damages, claims, costs, lost profits or lost savings.

2.8 Indemnity

You agree to defend, indemnify, and hold harmless the copyright holder and its parents and affiliates and all of their employees, agents, directors, officers, proprietors, partners, representatives, shareholders, servants, attorneys, predecessors, successors, assigns, and those who have worked on the preparation, publication or distribution of the **S3000L™ suite of information** from and against any and all claims, proceedings, damages, injuries, liabilities, losses, costs, and expenses (including reasonable attorneys' fees and litigation expenses), relating to or arising from your use of the **S3000L™ suite of information** or any breach by you of this user agreement.

2.9 Governing law and arbitration

This user agreement will be governed by and construed in accordance with the laws of the Kingdom of Belgium.

In the event of any dispute, controversy or claim arising out of or in connection with this user agreement, or the breach, termination or invalidity thereof, the parties agree to submit the matter to settlement proceedings under the ICC (International Chamber of Commerce) ADR rules. If the dispute has not been settled pursuant to the said rules within 45 days following the filing of a request for Alternative Dispute Resolution (ADR) or within such other period as the parties may agree in writing, such dispute shall be finally settled under the rules of arbitration of the International Chamber of Commerce by three arbitrators appointed in accordance with the said rules of arbitration. All related proceedings should be at the place of the ICC in Paris, France.

The language to be used in the arbitral proceedings shall be English.

3 Special usage rights

Permission to use, reference, or deliver training from the information contained in this document, and in the S-Series IPS Specifications, and the right to reproduce or publish the S-Series IPS Specifications, in whole or in part, is hereby given to the following:

1 National Associations who are members of ASD and all their member companies.

2 Members of Aerospace Industries Association of America.

3 Armed Forces that are customers of Companies included in these special usage rights Categories 1 and 2 inclusively.

4 Ministries and Departments of Defense of the member countries of ASD, AIA, NATO, and NATO Partners.

5 NATO bodies, organizations & agencies.

6 The industries of ASD, AIA, NATO's nations, and NATO's Partners.

7 Universities, Educational Institution, Technologies and Research Institutes within ASD, AIA, NATO nations and NATO Partners.

Highlights

This issue includes updates to the Copyright, user agreement and special usage rights only. There have been no technical changes in this issue. The highlights below are those of the 2021 Block Release.

Table of contents

List of tables

Table 1 References

Chap No./Document No.	Title
S1000D	International specification for technical publications using a common source database
S4000P	International specification for developing and continuously improving preventive maintenance
S6000T	International specification for training analysis and design
SX000i	International specification for Integrated Product Support (IPS)

SX002D	Common data model for the S-Series IPS Specifications
SX004G	Unified Modeling Language (UML) model reader's guidance

1 General

Changes that are included in this issue 2.0 of S3000L are results from improvements in existing processes and the introduction of new content in several chapters. Additionally, the S3000L data model was updated in line with the common data model of the S-Series IPS Specifications, which is documented in SX002D.

2 Technical and editorial changes

Technical and editorial changes are not change marked in this issue 2.0 of S3000L. Any editorial changes are not described in the highlight pages.

3 Content

Highlights of changes to chapters 1 to 22 are provided in the following tables:

- chapter 1 - Introduction to the specification, refer to Table 2
- chapter 2 - General requirements, refer to Table 3
- chapter 3 - LSA process, refer to Table 4
- chapter 4 - Product structures and change management in LSA, refer to Table 5
- chapter 5 - Influence on design, refer to Table 6
- chapter 6 - Human factors analysis, refer to Table 7
- chapter 7 - Corrective maintenance analysis, refer to Table 8
- chapter 8 - Damage and special event analysis, refer to Table 9
- chapter 9 - Operational support analysis, refer to Table 10
- chapter 10 - Development of a preventive maintenance program, refer to Table 11
- chapter 11 - Level of repair analysis, refer to Table 12
- chapter 12 - Task requirements and maintenance task analysis, refer to Table 13
- chapter 13 - Software support analysis, refer to Table 14
- chapter 14 - Life cycle cost considerations, refer to Table 15
- chapter 15 - Obsolescence analysis Table 16
- chapter 16 - Disposal, refer to Table 17
- chapter 17 - In-service LSA, refer to Table 18
- chapter 18 - Interrelations to other S-Series IPS specifications, refer to Table 19
- chapter 19 - Data model, refer to Table 20
- chapter 20 - Data exchange, refer to Table 21
- chapter 21 - Terms, abbreviations and acronyms, refer to Table 22
- chapter 22 - Data element list, refer to Table 23

Table 2 Introduction to the specification

Chapter	Summary of changes
Chap 1	– clarification added regarding S3000L compliance and logistics support vs. logistic support

Table 3 General requirements

Chapter	Summary of changes
Chap 2	– term "LSA Guidance Conference Document (LSA GCD)" was changed to "LSA Guidance Document (LSA GD)" in general – term "implementation rules" was changed to "business rules" in the LSA GD – term "scheduled maintenance" was changed to "preventive maintenance" in general (in some cases, the term scheduled maintenance is kept in the context of preventive maintenance tasks which are able to schedule based on repeated intervals) – reference to logistic/logistics was replaced by product support – additional paragraph listing the relevant Unit of Functionality (UoF) from Chap 19 for the population of LSA data which are an outcome of Chap 2

Table 4 LSA process

Chapter	Summary of changes
Chap 3	– general clarification of the further usage of the terms "logistics" and "supportability" within the specification – improvement and completion of relations to the IPS elements – improvement and completion of content concerning event driven corrective and preventive maintenance in the context of FMEA, special event analysis and Preventive Maintenance Analysis (PMA), including the harmonization of wording according to S4000P – adaption of wording to harmonize with Chap 19 – additional paragraph listing the relevant UoF from Chap 19 for the population of LSA data which are an outcome of Chap 3

Table 5 Product structures and change management in LSA

Chapter	Summary of changes
Chap 4	– chapter was completely reworked to clearly explain the full capabilities for the generation of different product breakdown structures, including breakdown element realization, part lists and product variant aspects – establishing a logical chapter structure based on Units of Functionality (UoF) of Chap 19 enabling traceability with the S3000L data model – rework of change management aspects for a better understanding of how S3000L supports change management processes and how it impacts the LSA process – generating additional illustrations and rework of existing illustrations from S3000L, issue 1.1, for a better understanding of product breakdown structure and change management concepts – additional paragraph listing the relevant UoF from Chap 19 for the population of LSA data which are an outcome of Chap 4

Table 6 Influence on design

Chapter	Summary of changes
Chap 5	– no major technical changes in Chap 5, only editorial rework – additional paragraph listing the relevant UoF from Chap 19 for the population of LSA data which are an outcome of Chap 5

Table 7 Human factors analysis

Chapter	Summary of changes
Chap 6	– no major technical changes in Chap 6, only editorial rework

Table 8 Corrective maintenance analysis

Chapter	Summary of changes
Chap 7	– clarification added about the objectives and perimeter of the chapter – separation of the main paragraph into two separate paragraphs to distinguish corrective maintenance on the Product and corrective maintenance on equipment at workshop after it has been removed from the Product – clarification added about the type of input (FMEA) and the process in each case – additional paragraph listing the relevant UoF from Chap 19 for the population of LSA data which are an outcome of Chap 7

Table 9 Damage and special event analysis

Chapter	Summary of changes
Chap 8	– separation of special event analysis and damage analysis – clarification of special event process to be consistent and complementary to corresponding paragraphs dedicated to special events in S4000P – clarification of damage analysis as an independent process with the following steps: identification of damage with different inputs, assessment of damage relevance, analysis of acceptable damage size and repair procedure – additional paragraph listing the relevant UoF from Chap 19 for the population of LSA data which are an outcome of Chap 8

Table 10 Operational support analysis

Chapter	Summary of changes
Chap 9	– no major technical changes in Chap 9, only editorial rework – additional paragraph listing the relevant UoF from Chap 19 for the population of LSA data which are an outcome of Chap 9

Table 11 Development of a preventive maintenance program

Chapter	Summary of changes
Chap 10	– harmonization of terminology with S4000P, focusing on different classes of Preventive Maintenance Task Requirements (PMTR), Preventive Maintenance Task Requirement Interval (PMTRI) for a PMTR based on an interval and Preventive Maintenance Task Requirement Event (PMTRE) for PMTR after the occurrence of a special event – harmonization of terminology with Chap 19 in the context of time limits, thresholds and triggers – inclusion of an interval adaption example in the context of preventive maintenance packaging – additional paragraph listing the relevant UOF from Chap 19 for the population of LSA data which are an outcome of Chap 10

Table 12 Level of repair analysis

Chapter	Summary of changes
Chap 11	– clarification of the term maintenance level in the context of personnel competence, facilities or support equipment available – inclusion of an additional sub paragraph on the definition of operational concept as part of data gathering for Level of Repair Analysis (LORA) – additional paragraph listing the relevant UoF from Chap 19 for the population of LSA data which are an outcome of Chap 11

Table 13 Task requirements and maintenance task analysis

Chapter	Summary of changes
Chap 12	– extension and improvement of the explanation of the task requirements approach (each rectifying task needs to be justified by at least one task requirement) – improvement of the presentation of the task categorization – rework of the task documentation paragraphs referring to the representation of the UoF (refer to Chap 19) covering the aspects of Maintenance Task Analysis (MTA) – adding a paragraph to describe the data model capability to document required circuit breaker settings to perform a task – complete rework of maintenance task examples (deletion of misleading figures in the paragraph) – extended adaption of wording to harmonize with Chap 19 – additional paragraph listing the relevant UoF from Chap 19 for the population of LSA data which are an outcome of Chap 12

Table 14 Software support analysis

Chapter	Summary of changes
Chap 13	– improvement of the explanation, which software related tasks are relevant and to be included in LSA activities from a supportability point of view – adaption of wording to harmonize with the data model chapter 19 – additional paragraph listing the relevant UoF from Chap 19 for the population of LSA data which are an outcome of Chap 13

Table 15 Life cycle cost considerations

Chapter	Summary of changes
Chap 14	– inclusion of the synchronization of LCC analysis between contractor and customer software tools – additional paragraph listing the relevant UoF from Chap 19 for the population of LSA data which are an outcome of Chap 14

Table 16 Obsolescence analysis

Chapter	Summary of changes
Chap 15	– reorganization of the paragraphs for better readability – alignment of the phases definitions with SX000i and relevant chapters/paragraphs – inclusion of additional concepts and references (eg, Registration, Evaluation, Authorization and Restriction of Chemicals (REACH)) – additional paragraph listing the relevant UoF from Chap 19 for the population of LSA data which are an outcome of Chap 15

Table 17 Disposal

Chapter	Summary of changes
Chap 16	– former chapter number 17 from S3000L, issue 1.1, is now chapter number 16. In-service LSA chapter was shifted to chapter number 17, refer to Table 18. **Note** Disposal aspects are applicable during all life cycle phases of the product and provide task requirements. Logical sequence of the chapters was changed to have all procedural chapters providing task requirements before the In-service LSA chapter. – harmonization of terminology within chapter figures – additional paragraph listing the relevant UoF from Chap 19 for the population of LSA data which are an outcome of Chap 16

Table 18 In-service LSA

Chapter	Summary of changes
Chap 17	– former chapter number 16 from S3000L, issue 1.1, is now chapter number 17. In-service LSA chapter was shifted to chapter number 17, refer to Table 17. – complete rework of in-service LSA chapter – rework of definition of in-service LSA activities – inclusion of flowcharts and explaining descriptions to start an In-Service Support Optimization (ISSO) process – definition of ISSO analysis process subdivided by different support task types, including flowcharts and basic analysis questions for each support task type – inclusion of a detailed ISSO analysis logic to analyze a disposal task as an example **Note** The example includes detailed flowchart diagrams as an example how to prepare such analysis baselines within an ISSO guidance document. – rework of ISSO follow on phase – additional paragraph listing the relevant UoF from Chap 19 for the population of LSA data which are an outcome of Chap 17

Table 19 Interrelations to other S-Series IPS specifications

Chapter	Summary of changes
Chap 18	– restructure of chapter paragraphs to provide a better overview – S-Series IPS Specifications completed by adding S6000T – rework of the interface description to engineering and supportability engineering

Table 20 Data model

Chapter	Summary of changes
Chap 19	– removal of introduction paragraphs on how to read UML models **Note** The content of the introduction about how to read UML models is covered by the specification SX004G.

Chapter	Summary of changes
	The part of Chap 19 which contains the graphical representations and descriptions of the respective Units of Functionality (UoF) has been reworked. Many changes were driven by the harmonization with SX002D, issue 2.1. In this context, changes were incorporated like: – updated representations of classes, attributes and relationships (including cardinality) inside many UoF – renaming of classes or attributes – separation of specific aspects within former UoF of S3000L (issue 1.1) to their own UoF – splitting of some UoF into several UoF to simplify and improve readability of the data model **Note** Detailed changes like renaming, changes of classes and attributes, changes of relationships, changes of cardinalities, changes in graphical representation, editorial changes, which were incorporated to the single UoF are not mentioned in this highlights chapter. Additionally, new UoF were created to support new subjects from the procedural chapters 2 to 17.
	In the following listing, a general overview concerning changes is given for each single UoF of S3000L, issue 2.1 – UoF Aggregated Element Incorporation of minor editorial changes. – UoF Applicability Statement Incorporation of minor changes to simplify the reading and implementation of the data model. Incorporation of changes to harmonize with SX002D, issue 2.1. – UoF Breakdown Structure Incorporation of changes to harmonize with SX002D, issue 2.1. The one significant change that has been introduced is the possibility to define relationships between breakdown revisions (BreakdownRevisionRelationship). – UoF Change Information New UoF to allow for traceability of authorized changes. This change is harmonized with SX002D, issue 2.1. – UoF Circuit Breaker New UoF for the separated representation of circuit breakers. This change is driven by the harmonization with SX002D, issue 2.1. – UoF Competence Definition New UoF for the separated representation of competence definitions. This UoF includes a portion of the previous UoF Task Resources in S3000L, issue 1.1. Changes are driven by the harmonization with SX002D, issue 2.1, and S6000T issue 2.0.

Chapter	Summary of changes
	– UoF Damage Definition New UoF for the separated representation of damages. This UoF includes a portion of the previous UoF Special Event and Damage in S3000L, issue 1.1. This change is harmonized with SX002D, issue 2.1 and has been part of the harmonization with S4000P. – UoF Decision Tree Template Definition New UoF to support in-service LSA. Changes are driven by the completely reworked content of the in-service LSA, Chap 17. This change is harmonized with SX002D, issue 2.1. – UoF Design Change Request New UoF for the separated representation of a design change request. This UoF includes a portion of the previous UoF LSA Candidate Task Requirement in S3000L, issue 1.1. This change is driven by the harmonization with SX002D, issue 2.1. – UoF Digital File New UoF to support the exchange of digital files as part of the data set. This change is harmonized with SX002D, issue 2.1. – UoF Document Incorporation of minor changes to harmonize with S1000D. The one significant change that has been introduced is that any class in the data model now can define references to documents. This change is harmonized with SX002D, issue 2.1. – UoF Environment Definition New UoF to add capabilities for the definition of concepts included in the UoF Product Usage Context, for task definitions and applicability statements. This change is harmonized with SX002D, issue 2.1. – UoF Facility New UoF for the separated representation of facilities and includes portions of the previous UoF Product Usage Context in S3000L, issue 1.1. This change is harmonized with SX002D, issue 2.1. – UoF Failure Mode New UoF for the separated representation of failure modes. This UoF includes a portion of the previous UoF LSA Failure Mode in S3000L, issue 1.1. Changes are driven by harmonization with S4000P and modifications within Chap 7 how to use FMEA data in the context of corrective maintenance analysis. – UoF Failure Mode Isolation New UoF for the separated representation of failure modes. This UoF includes a portion of the previous UoF LSA Failure Mode in S3000L, issue 1.1. Changes are driven by harmonization with S4000P and modifications within Chap 7 how to use FMEA data in the context of corrective maintenance analysis. – UoF Failure Mode Symptom New UoF for the separated representation of failure modes. This UoF includes a portion of the previous UoF LSA Failure Mode in S3000L, issue 1.1. Changes are driven by harmonization with S4000P and modifications within Chap 7 how to use FMEA data in the context of corrective maintenance analysis.

Chapter	Summary of changes
	– UoF Hardware Element New UoF for the separated representation of a hardware element. Changes are driven by the harmonization with SX002D, issue 2.1. – UoF In Service Optimization Analysis New UoF to support in-service LSA. Changes are driven by the completely reworked content of the in-service LSA, Chap 17. This change is harmonized with SX002D, issue 2.1. – UoF Location New UoF to provide additional capabilities for the definition of an operating base and a maintenance facility. This change is driven by the harmonization with SX002D, issue 2.1. – UoF Logistics Support Analysis Message Content New UoF to support data exchange. Creation of UoF is driven by harmonization with SX002D, issue 2.1, and by the need to define the structure for the S3000L XML data set. – UoF LSA Candidate New UoF to replace the previous UoF LSA Candidate Analysis Activity and portions of the previous UoF LSA Candidate Item in S3000L, issue 1.1. Changes are driven by the harmonization with SX002D, issue 2.1. – UoF LSA Failure Mode Group New UoF for the separated representation of failure modes. This UoF includes a portion of the previous UoF LSA Failure Mode in S3000L issue 1.1. Changes are driven by harmonization with S4000P and modifications within Chap 7 how to use FMEA data in the context of corrective maintenance analysis. – UoF Message New UoF to support data exchange. Creation of UoF is driven by harmonization with SX002D, issue 2.1. – UoF Organization Assignment The one significant change that has been introduced is that any class in the data model now can define references to organizations. This change is harmonized with SX002D, issue 2.1. – UoF Part Definition Incorporation of additional classes/attributes and changes of attribute names. Changes are driven by harmonization with SX002D, issue 2.1. The one significant change that has been introduced is that parts lists now have explicit revisions. – UoF Performance Parameter New UoF to replace portions of the previous UoF LSA Candidate in S3000L, issue 1.1. Creation of UoF is driven by harmonization with SX002D, issue 2.1. – UoF Product and Project Incorporation of minor changes. Changes are driven by the harmonization with SX002D, issue 2.1.

Chapter	Summary of changes
	– UoF Product Design Configuration Representation of this UoF was modified significantly. Changes are driven by the need to improve readability of the UoF and a better representation of Product configuration aspects within Chap 4. This change is harmonized with SX002D, issue 2.1. – UoF Product Usage Context Incorporation of minor changes. Changes are driven by the harmonization with SX002D, issue 2.1. – UoF Product Usage Phase New UoF for the separated representation of usage phases. This UoF includes a portion of the previous UoF Special Event and Damage in S3000L, issue 1.1. Creation of UoF is driven by harmonization with SX002D, issue 2.1. – UoF Remark The one significant change that has been introduced is that any class and attribute in the data model can have remarks. This change is harmonized with SX002D, issue 2.1. – UoF Resource Specification New UoF for the separated representation of resource specification aspects. This UoF includes a portion of the previous UoF Task Resource in S3000L, issue 1.1. This change is harmonized with SX002D, issue 2.1. – UoF Security Classification Incorporation of minor changes. Changes are driven by the harmonization with SX002D, issue 2.1. – UoF Software Element New UoF for the separated representation of a software element. Changes are driven by the harmonization with SX002D, issue 2.1. – UoF Special Event New UoF to replace portions of the previous UoF Special Event and Damage in S3000L, issue 1.1. Changes are driven by the need to have a better representation of special events as described in Chap 8 in the S3000L data model. This change is harmonized with SX002D, issue 2.1. – UoF Task Incorporation of minor changes. Aspect of circuit breaker definition was moved to its own UoF. Changes are driven by the harmonization with SX002D, issue 2.1. – UoF Task Requirement New UoF to replace portions of the previous UoF LSA Candidate Task Requirement in S3000L, issue 1.1. Changes are mainly driven by the harmonization with SX002D, issue 2.1. – UoF Task Resource Incorporation of minor changes to simplify the reading and implementation of the data model, adding of attributes and moving resource specification to a separate UoF. Changes are mainly driven by the harmonization with SX002D, issue 2.1.

Chapter	Summary of changes
	– UoF Task Usage New UoF to combine portions of the previous UoF Task Usage (Part 1) with the previous UoF Task Usage (Part 2) in S3000L, issue 1.1. Incorporation of renaming of attributes and minor modification of classes. Changes were driven by the need to have a logic separation of different aspects which were included in one UoF in S3000L, issue 1.1. Changes are harmonized with SX002D, issue 2.1. – UoF Time Limit New UoF for the separated representation of time limits and thresholds. This UoF includes a portion of the previous UoF Task Usage (Part 1) in S3000L, issue 1.1. UoF is extended by additional attributes and relations. Adaption also includes the capability to handle condition based maintenance in the UoF. Changes are harmonized with SX002D, issue 2.1. – UoF Zone Element Incorporation of minor changes. A breakdown element (or a revision thereof) can now be associated with a zone. Changes are driven by the harmonization with SX002D, issue 2.1.

Table 21 Data exchange

Chapter	Summary of changes
Chap 20	– no major technical changes in Chap 20, only editorial rework **Note** The exchange of data for S3000L is defined using XML and XML schema. S3000L XML schemas are published separately on the S3000L website (www.s3000l.org).

Table 22 Terms, abbreviations and acronyms

Chapter	Summary of changes
Chap 21	– update of terms, abbreviations and acronyms according to the modifications performed for all chapters

Table 23 Data element list

Chapter	Summary of changes
Chap 22	– update of data elements and data element descriptions according to the changes established within Chap 19 – inclusion of a list of all classes defined in the respective UoF in Chap 19 and the corresponding class definitions – inclusion of a list which contains valid values for classification attributes

Table of contents

The listed documents are included in issue 2.1, dated 2023-03-01, of this publication.

Chapter 1

Introduction to the specification

Table of contents

List of tables

List of figures

References

Table 1 References

Chap No./Document No.	Title
Chap 1	Introduction to the specification
Chap 2	General requirements
Chap 3	LSA process
Chap 4	Product structures and change management in LSA
Chap 5	Influence on design
Chap 6	Human factors
Chap 7	Corrective maintenance analysis
Chap 8	Damage and special event analysis
Chap 9	Operational support analysis
Chap 10	Development of a preventive maintenance program
Chap 11	Level of repair analysis
Chap 12	Task requirements and maintenance task analysis
Chap 13	Software support analysis
Chap 14	Life cycle cost considerations
Chap 15	Obsolescence analysis
Chap 16	Disposal
Chap 17	In-service LSA
Chap 18	Interrelations to other S-Series IPS specifications
Chap 19	Data model
Chap 20	Data exchange
Chap 21	Terms, abbreviations and acronyms
Chap 22	Data element list
S1000D	International specification for technical publications using a common source database
S2000M	International specification for material management - Integrated data processing
S4000P	International specification for developing and continuously improving preventive maintenance

S5000F	International specification for in-service data feedback
S6000T	International procedure specification for training analysis and design
SX000i	International specification for Integrated Product Support (IPS)
IPS-C-2020-010	Governance of the S-Series Specifications
DEF-STAN-00-60	Integrated Logistic Support - UK MoD
DEX1A&D	DEX1A&D - Aerospace and defense business DEX for exchange of product breakdown for support
DEX3A&D	DEX3A&D - Aerospace and defense business DEX for exchange of a task specification
ISO 10303-239 PLCS	Product Life Cycle Support (PLCS)
MIL-HDBK-502	Department of defense handbook acquisition logistics
MIL-STD-1388-1A	Logistics support analysis - DoD
MIL-STD-1388-2B	DoD requirements for a logistic support analysis record

1 General

This chapter gives a basic overview of the purpose of the S3000L specification, including its development through the years.

1.1 Purpose

Logistics Support Analysis (LSA) is one of the most important processes of Product support.

It is the main tool for:

- designing Products relevant to maintainability, reliability, and testability activities, and to optimize life cycle cost
- defining all required resources to support the Product in its intended use during in-service operation

S3000L defines the processes, general requirements and related information exchange governing the performance of LSA during the life cycle of aerospace, defense, and commercial Products.

1.2 Background

The development of this specification originated within the Aerospace and Defence Industries Association of Europe (ASD) in 2005. At that time, MIL-STD 1388-1A was no longer in force and DEF STAN 00-60 had become too specific to UK-MoD acquisitions to be applied as an international handbook for general purposes.

The lack of a common valid procedure resulted in the development of various project specific LSA handbooks and relevant IT solutions. These specific activities were also needed for new projects (such as Eurofighter Typhoon, NH-90, Tiger, A400M, and Gripen). This caused considerable effort for both industry and its military and non-military customers.

This situation prompted ASD and the Aerospace Industry Association of America (AIA) to consider the joint development of a new common international specification for LSA.

This initiative started with an "Inaugural meeting on S3000L" held in Brussels on January 18, 2006. The attendees expressed their thoughts on the need for an international LSA specification in the light of shortfalls in current standards, specifications, and handbooks. It emerged that it was not a matter of shortfalls, but rather an overabundance of standards.

In March 2006, in Munich, the following basic requirements were identified during a meeting for the project definition phase. The new LSA specification will:

- be generally based on the processes laid out in MIL-STD-1388-1A, MIL-HDBK-502, DEF-STAN-00-60 and the activity model given by ISO 10303-239 PLCS
- be the standard for creation and development of LSA data exchanged by:
  - DEX1A&D - Aerospace and defense business DEX for exchange of Product breakdown for support
  - DEX3A&D - Aerospace and defense business DEX for exchange of a task specification
  - Another exchange mechanism to address the relevant data from MIL-STD-1388-2B for LSA purposes.
- use experience gained from the performance of LSA for current programs such as Eurofighter Typhoon, NH-90, Tiger, A400M, Rafale, Gripen, JSF, etc
- include process application guidelines and rules for information exchange
- be tailorable to specific needs and include guidelines for tailoring
- take into account current ISO/EN baseline documents
- enable interfaces to the suite of the S-Series IPS specifications: S1000D, S2000M, S4000P, S5000F and S6000T

An international team of experts, working under the joint supervision of AIA and ASD representatives, contributed to the development of the specification. Table 2 shows the companies/organizations contributed to the initial phases of the activity.

Table 2 Participating companies/organizations

Company	Country
AgustaWestland	United Kingdom
Airbus Deutschland GmbH	Germany
Boeing	United States
Dassault Aviation	France
EADS Casa	Spain
EADS Military Air Systems	Germany
Eurocopter	France
LOGSA	United States
MBDA	France
Saab AB	Sweden

The final draft of S3000L (Issue 0.1) was officially published in June 2009. The main purpose of this draft was to enable experts from interested companies and organizations to provide the S3000L expert team with comments on the first approach. The commenting phase officially concluded at the end of 2009. In this phase, experts from several nations contributed their input to improve the final draft for the publication of the first official Issue 1.0.

In June 2010, Issue 1.0 of S3000L was finally released and published. At the Farnborough Air Show in July 2010, ASD and AIA signed a Memorandum of Understanding to form the ASD/AIA Integrated Product Support (IPS) Council. The IPS community implemented a new platform to harmonize and coordinate the different IPS specification activities.

S3000L Issue 1.1 is the consequence of the constant review to maintain and improve the specification and harmonize it with other specifications included in the S-Series IPS Specifications. Additionally, XML schema became the new format for data exchange. In July 2014, S3000L Issue 1.1 was released and published together with the XML schemas.

S3000L Issue 2.0 is the consequence of various programs implementing S3000L and providing feedback. Furthermore, there is an ongoing harmonization process with other specifications of the S-Series IPS Specifications.

1.3 Scope

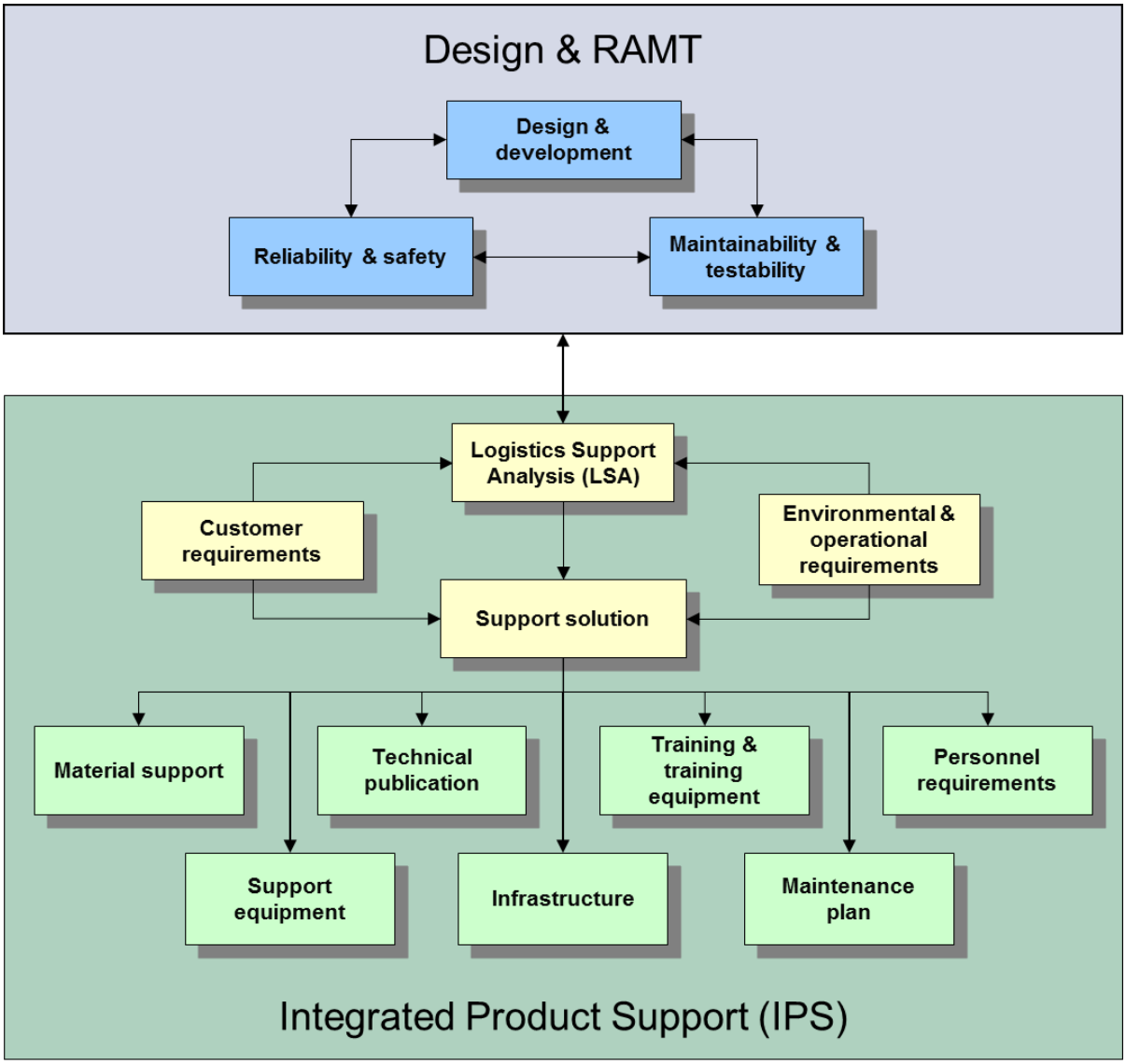
S3000L design covers all processes and requirements governing the performance of LSA activities.

- It provides rules for the usage of Product breakdown for Product support purposes and for the selection of LSA candidate items
- It describes the types and methods to perform specific analyses
- It gives guidelines on how to process the results of the analyses and achieve a cost-effective support solution
- It covers the interface between LSA and the support engineering areas, for example Reliability, Availability, Maintainability and Testability (RAMT)
- It covers the interface between LSA and the IPS functional areas such as supply support, technical data, special tools/test equipment (support equipment), and training. Refer to Fig 1.

In particular, S3000L describes the interaction between industry (contractor) and the customer that, as per the existing agreements, will provide the typical deliverables of LSA as given in this specification. Examples of typical deliverables of an LSA process include, but are not limited to:

- a reliable and maintainable Product because of influence on design from a Product support perspective
- cost efficient support system
- supportability product data
- support concept in the context of identification and description of maintenance tasks (corrective and preventive) and operational support tasks

S1000D and S2000M specifications describe in detail activities related to technical documentation and material management, respectively. These activities are therefore out of S3000L scope. This also applies to methodologies on carrying out Preventive Maintenance Analysis (PMA) as described in S4000P or detailed performance of Level of Repair Analysis (LORA).



ICN-B6865-S3000L0001-003-01

Fig 1 Functional elements of integrated product support

2 How to use the specification

2.1 General

This chapter gives an overview of:

- how to apply S3000L to a project
- the organization of the specification
- the fundamental reading rules

2.2 Application

2.2.1 S3000L application policy

S3000L will be the common LSA specification for customers and contractors (eg, governments, procurement authorities, support agencies, and industry). Customers and contractors can agree to integrate S3000L with additional international or national requirements for specific projects. The use of the specification and any additional processes is the object of the agreement between the customer and contractor.

2.2.2 S3000L application

At the start of any project using S3000L, it is necessary for the customer and contractor to agree on how the specification will be implemented, and to define and agree on the variables and options that S3000L provides. The project's LSA Guidance Document (LSA GD) must contain these decisions. The customers and contractors use the LSA Guidance Conference (LSA GC) to determine the information to be included in the LSA GD. Refer to Chap 3.

It is necessary to define a project interchange agreement, for data and other deliverables, in order to supplement the LSA GD. Based on the project complexity, the specification will either be stand-alone, or integrated within the LSA GD.

2.2.3 Tailoring processes

In order to ensure efficient implementation, the S3000L design helps users select the functionality that is appropriate to their specific projects.

Individual chapters can be included or excluded. Tailoring can also determine the number and range of LSA candidate items, the number of analyses per candidate item and the data used during an analysis.

2.2.4 Compliance with S3000L

ASD and AIA do not provide any type of validation that a software product, a training course, or any other product is compliant with the S3000L specification. This is also valid for compliance with the methodology of the proposed analysis processes.

2.3 Basic definitions

Product	Any platform, system or equipment (air, sea, land vehicle, equipment or facilities, civil or military).
Product variant	A member of a Product family that is configured for a specific purpose and is offered to customers. The Product Variant is uniquely identified by the *Model Identifier.* The term "Product Variant" is synonymous with "model."
Project	The task to develop, maintain and dispose of the Product.

2.4 Integrated Logistics Support vs. Integrated Product Support

In the context of product support, these terms are treated like synonyms. On the other hand, "logistic" is often used in the context of the transportation business. Beginning in this issue of the S3000L specification, the term Integrated Product Support (IPS) will be used to avoid confusion.

Note

The official name for the processes and analyses commonly known as Logistics Support Analysis will continue to be known as LSA.

3 Organization of the specification

S3000L is organized into chapters, each dealing with individual aspects of LSA.

3.1 Chapter 1 - Introduction to the specification

Chap 1 provides a summarized view on purpose, background, scope, and application of S3000L. It also illustrates the rules regarding the maintenance of S3000L and explains how to access the specification.

3.2 Chapter 2 - General requirements

Chap 2 provides general information on how to set up and manage an LSA program as part of an IPS and overall development program. It describes the interfaces, dependencies, and time constraints in relation to the other IPS disciplines within a typical project scenario.

3.3 Chapter 3 - LSA process

Chap 3 is designed to provide a full, in-depth description of the complete LSA process. It covers the activities beginning at the definition of the operational framework to the starting point of the creation of IPS element deliverables (refer to SX000i). The operational framework and contractual requirements provide input for the LSA GD, which establishes the binding business rules for the LSA process. This includes establishing a Product breakdown method, rules for LSA candidate selection, and tailoring the specification to the specific project needs.

The chapter further describes potential supportability analysis activities and documentation of analysis results within LSA data. Finally, it covers aspects concerning customer involvement during the LSA process and LSA review conference organization. This is the final step in accepting the LSA results and determining the maintenance and operational support concept. The acceptance of the LSA results can be regarded as a precondition to start the production of IPS element deliverables.

3.4 Chapter 4 - Product structure and change management in LSA

The generation of a Product breakdown is an essential activity of the LSA process. An appropriate Product breakdown ensures proper identification and versioning of the Product configuration throughout the Product life cycle. The chapter describes the different sources, methods, and criteria used to generate a Product breakdown.

It is also essential to control changes, analyze the impact on LSA, and record the implementation of changes in the Product physical and functional characteristics. Change management allows updating LSA for different Product baselines throughout the Product life cycle as consequence of different sources of Product configuration change (eg, triggered by design, technology, obsolescence, etc).

LSA plays an active role in the Product configuration management process to perform these activities as part of the LSA process, generation of a Product breakdown, and change management. This process records, communicates, and controls integrity and continuity of design, systems engineering, and supportability. Configuration management efforts provide a complete audit traceability of decisions and design modifications.

Chap 4 defines the details and methodology to apply the principles of Product breakdown generation and change management to the LSA process as part of the configuration management.

3.5 Chapter 5 - Influence on design

Influence on design is the goal of an industrial activity known as supportability engineering. Supportability engineering influences Product characteristics vital for Product operation according to performance and design requirements, with a view to minimizing Life Cycle Cost (LCC).

The Product characteristics to be influenced are primarily RAMT. Early RAMT activities/programs and the performance of LSA serve as tools for supportability engineering.

Design reviews are also encouraged after the Product entry into service, to improve the Product based on operational experience and an effective sustainment of a competitive Product. During the in-service phase of a Product, in-service data feedback provides a basis for measuring and analyzing the RAMT and LCC performance of the design.

Supportability engineering acts as the enabler between design/development and IPS.

Chap 5 describes how the methods of supportability engineering are applied and how the results are linked to the LSA process.

3.6 Chapter 6 - Human factors

Chap 6 describes the relationship and integration between human factors engineering and the support analysis process. Like all other supportability engineering functions, human factors analysis provides source data used by the support analysis team to determine maintenance crew and support equipment requirements. This relationship begins in the design review process and continues through the development of the Maintenance Task Analysis (MTA).

3.7 Chapter 7 - Corrective maintenance analysis

Chap 7 covers the methodology and decision logic applied to identify corrective maintenance task requirements, as well as the diagnosis procedure in case of Product failure during normal use. The scope includes corrective maintenance to be carried out on the Product to restore its availability as well as corrective maintenance of equipment removed from the Product in a workshop.

3.8 Chapter 8 - Damage and special event analysis

Chap 8 covers the methodology for identifying and justifying appropriate maintenance tasks in case of special events or damages found on the Product during general or detailed inspections.

3.9 Chapter 9 - Operational support analysis

Chap 9 covers the methodology for identifying and justifying tasks related to Product operation and operational support that cannot be performed in the areas where the Product is directly used (documented in operating instructions) or maintained (documented in a maintenance manual). However, they are performed in operations and maintenance facilities in support of normal operation (eg, packaging, handling, storage, mooring, preparation for transportation).

These tasks can be important for the proper use of any Product. There are many operational aspects (eg, ease of operation, usability, flexibility of usage, mobility) restricted by operational support tasks.

3.10 Chapter 10 - Development of a preventive maintenance program

Chap 10 describes how to handle Preventive Maintenance Task Requirements Interval (PMTRI) identified and determined by Preventive Maintenance Analysis (PMA) activities (refer to S4000P). Preventive maintenance is vital to the operation of Products. After identifying all PMTRI, the development of a preventive maintenance program is required, which results in preventive maintenance task packages performed at a specific time limit or interval.

The chapter further describes how to link corresponding preventive maintenance tasks to the PMTRI and to group the PMTRI/tasks based on appropriate time limits (eg, intervals based on Product usage). For this purpose, a series of rules indicates how to adapt PMTRI by extending or reducing the single PMTRI time limits during the grouping process.

3.11 Chapter 11 - Level of repair analysis

Chap 11 describes Level of Repair Analysis (LORA) activities performed as part of any acquisition program. An LSA plan or similar documents, usually include a request for LORA.

LORA is an analytical methodology performed on a list of selected LSA candidates. It takes into consideration customers' support capabilities and support/operational requirements, product technical information, costs, legal and environmental constraints to determine an optimized support solution. The outcome is the definition of where each selected candidate is for example removed, replaced, repaired, checked, overhauled, inspected, or discarded.

3.12 Chapter 12 - Task requirements and maintenance task analysis

Chap 12 covers two main aspects of the LSA process. One aspect explains how to combine a task requirement with the corresponding maintenance (corrective and preventive) or operational support task. The other main aspect analyzes the identified corrective maintenance, preventive maintenance, and operational support tasks in terms of task execution and required resources like spare parts, consumables, support equipment, personnel, facilities and infrastructure. Finally, additional information for tasks is included in the LSA data. Such information also includes preferred maintenance level, task duration, training requirements and criticalities with respect to personnel.

3.13 Chapter 13 - Software support analysis

Chap 13 provides guidelines on how to handle the requirements concerning software in the context of Product operation and maintenance. Today, software is an essential element of nearly all Products. The chapter describes the typical task requirements for software, as well as the relevant maintenance or operational support tasks. In this context, a clear differentiation is made between maintenance or operational support tasks relevant to the software, and software modification activities (eg, bug fixing).

3.14 Chapter 14 - Life cycle cost considerations

LCC includes all direct costs and any indirect-variable cost associated with the procurement, operation, support, and disposal of the Product throughout its life cycle.

Chap 14 provides an overview of LCC aspects and indicates which information developed by an LSA process can be used to support LCC analysis activities, and vice versa.

3.15 Chapter 15 - Obsolescence analysis

Chap 15 describes the relationship between obsolescence analysis and the support analysis process.

In the design and development process, LSA is aimed at avoiding/controlling the use of components and material that are likely to be subject to obsolescence in the early operational phase. During the in-service phase, the analysis of obsolescence events will also include LSA, which will lead to the definition of economic alternatives with respect to maintenance and support plans.

3.16 Chapter 16 - Disposal

Chap 16 deals with the disposal of Products and Product components, as a result of their active and/or passive use. Although disposal of Products normally occurs during the last phase of the life cycle, when acquiring a Product, it is necessary to take into consideration disposal in the early phases of every program plan.

Adequate disposal activities can include, but not be limited to:

- professional disposal of equipment/components, structural parts or consumables
- destruction/neutralization of toxic substances
- recycling to ensure a sustainable development
- demilitarization of defense systems, to avoid weapons proliferation

3.17 Chapter 17 - In-service LSA

Chap 17 gives an overview of LSA activities recommended for the in-service phase of a Product. This chapter covers the regular update of LSA data and corresponding IPS elements as a consequence of changes in engineering design, and of usage and/or support scenarios on the operator's side. It also covers the In-Service Support Optimization (ISSO) process. This process provides a methodology to optimize maintenance and operational support tasks continuously based on the operator's feedback. Feedback information/data from the Product

user provides the opportunity to compare assumptions based on analysis activities to the actual in-service experience/data. Additionally, analysis and solving problems, which occurred during maintenance or operational support is facilitated.

To ensure continuous maintainability and operability, it is necessary to scrutinize the maintenance and operational support concept repeatedly during the Product life cycle.

3.18 Chapter 18 - Interrelation to other S-Series IPS specifications

Chap 18 addresses the benefits of using the S-Series IPS-specifications in conjunction with this specification. The chapter addresses LSA relationships to the following specifications:

- SX000i - International specification for Integrated Product Support (IPS) specifications
- S1000D - International specification for technical publications using a common source database
- S2000M - International specification for material management - Integrated data processing
- S4000P - International specification for developing and continuously improving preventive maintenance
- S5000F - International specification for in-service data feedback
- S6000T - International specification for training analysis and design

3.19 Chapter 19 - Data model

Chap 19 defines a coherent UML data model supporting the S3000L LSA process and its interaction with the related business processes. These processes can depend on data derived from LSA or from business processes that provide input to the LSA process. Additionally, the chapter defines all data elements used in the S3000L data model.

3.20 Chapter 20 - Data exchange

Chap 20 defines the standard technology used for the exchange of data that results from LSA activities as defined in S3000L. The exchange of data for S3000L is defined using XML and XML schema.

3.21 Chapter 21 - Terms, abbreviations and acronyms

Chap 21 provides a glossary of terms, abbreviations, and acronyms used within S3000L. It contains two different lists with items in alphabetical order:

- Glossary of terms
  A comprehensive list of terms used in this specification

- Abbreviations and acronyms
  Definitions of abbreviations and acronyms used in this specification

3.22 Chapter 22 - Data element list

Chap 22 lists all of the data elements used in the S3000L data model and in the S3000L data exchange specifications.

Data elements are listed alphabetically by name. The list contains:

- Data element name
- Data element data type
- Data element definition, including a textual definition and a list of valid values
- Class name identifies all classes of the S3000L data model where the data element is used
- Unit of Functionality identifies all Units of Functionality (UoF) of the S3000L data model where the data element is used

4 Maintenance of the specification

4.1 General information

S3000L is maintained by the S3000L Steering Committee (SC) operating under the supervision of the IPS Council. Both the S3000L SC and the IPS Council include representatives from AIA and ASD member companies and nations.

Technical issues related to S3000L can be raised using the change request application located at www.sx000i.org.

Technical issues can, in due course, become a Change Proposal (CP), both of which should be submitted with the understanding that any revisions to S3000L can affect one or more of the other S-Series IPS specifications, and that proposed changes are subject to international agreement from AIA and ASD member companies and nations.

Upon receipt of a CP, the S3000L SC will follow the change management process described in IPS-C-2020-010, that includes obtaining an agreement from the participating organizations prior to the publication of changes. The S3000L SC considers CP at each meeting and ratifies them for incorporation in the specification or otherwise. The S3000L SC also decides when changes will be published in S3000L.

Chapter 2

General requirements

Table of contents

List of tables

List of figures

References

Table 1 References

Chap No./Document No.	Title
Chap 10	Development of a preventive maintenance program
Chap 19	Data model
Chap 20	Data exchange
MIL-STD 1388-2B	DoD requirements for a logistic support analysis record
GEIA-STD-0007	Logistics Product Data

1 General

1.1 Introduction

The Logistics Support Analysis (LSA) process is considered a subset of Integrated Product Support (IPS). IPS has overall responsibility for the development of technical information and the support environment used to support a Product throughout its intended life cycle. It is necessary to harmonize the different disciplines in the context of supportability. The main disciplines are:

- design influence
- product support management
- supply support
- support and test equipment
- technical data/technical publication
- personnel and manpower
- IT/software support
- facilities and infrastructure
- maintenance planning
- preventive maintenance program
- Packaging, Handling, Storage and Transportation (PHST)
- training and training devices

The LSA program is the principal source of technical data for IPS planning and resource decision. LSA will be used:

- to link product design and IPS requirements to product readiness thresholds and to define detailed support element requirements
- throughout the acquisition cycle to assess and alter product design and to establish and update support element requirements
- as an important source of design-related data for determining and integrating all supportability requirements, analyzing alternative design, operational, and support concepts, and conducting trade-offs between design and the various elements of Product support of technical data for IPS planning and resource decision

The supportability analysis results and support resource data will be stored in a database to give all program elements access to a common data source for evaluating the supportability of design alternatives and objectives. The Integrated Product Team (IPT) will ensure the timely interaction of the LSA activities and data for all elements of the design process.

1.2 Purpose

IPS, supported by an embedded LSA process, ensures there is an open exchange of information to the product support disciplines and to the design. It pursues two major goals simultaneously:

- **Design for support**
  A focus on designs that minimize operation, maintenance, training, support tasks, and lifecycle costs while optimizing operational readiness.

- **Design of support**
  The design, development, funding, testing, and acquisition of all support resources needed to ensure optimum performance and readiness of the Product in its intended operational environment and usage/mission profiles.

1.3 Scope

This chapter is directed at IPS managers and LSA managers both for customers and for contractors. It is important to understand how an LSA program needs to be integrated into the superordinate IPS program. The approach for implementing an LSA program varies across organizations, and sometimes even across projects within an organization. Regardless of the organization or project, there must be shared policies that establish the functional responsibilities and expectations of the LSA program.

2 LSA program

LSA is a systematic, iterative process that integrates product design and support system requirements, and evaluates supportability requirements relative to satisfying the specific system/component sustainability and availability requirements. The use of an electronic database optimizes the analysis documentation process. Information on supportability consists in the identification, optimization, and traceability of IPS resources (eg, spares, manpower, personnel, support and test equipment, training and training equipment, facilities, PHST, and technical publication).

2.1 LSA program plan

2.1.1 Organizational requirements

The LSA Program Plan (LSA PP) describes the strategy for developing LSA activities during the engineering and manufacturing development phase of the complete program. It identifies and integrates the LSA activities, identifies management responsibilities and activities, and establishes the approach for analysis activities. Furthermore, this plan provides the approach to LSA and a basis to measure progress throughout the various phases of the program. Since the LSA process is iterative and dynamic, this plan will be updated to reflect current program status and planned changes.

This LSA PP describes the LSA process and the management structure established to perform this process. It details how the LSA process will take place to satisfy the intended requirements. LSA is an iterative process that usually continues as long as the item under analysis is in continuous use. Once the item has been retired, the LSA data can be used as baseline comparison data for future supported items.

The LSA PP describes the management, organizations, and procedures required to accomplish program requirements. It includes:

- a description, correlation, and schedule of procedures to be accomplished by each organizational element concerned in support of the LSA program

- an indication of requirements for LSA program participants (eg, subcontractors, suppliers, partner companies, agencies)
- the identification of LSA program monitoring points
- the dissemination of LSA requirements to design and description of techniques to be employed to ensure that desired LSA is inherent in the Product design
- a description of the planned approach for LSA activities, and specific methods and sources to satisfy qualitative supportability analysis and resolve design issues
- a description of numbering systems for database, Work Breakdown Structure (WBS), breakdown element identification, and handling methods for government furnished information
- a description of methods to identify, control, and report problems
- a description of provisions for plan update

Systems engineering managers are responsible for LSA planning and coordination. LSA managers provide technical guidance in conducting LSA and have special analytical skills required to perform activities specific to LSA. Procedures must ensure the identification of supportability requirements as an integral part of systems engineering and design. Supportability design baselines identify:

- quantified reliability and maintainability comparative analysis requirements, such as:
  - maintenance man-hours per operating hour
  - Mean Time Between Failure (MTBF)
  - Mean Time to Repair (MTTR)
- the maintenance concept, including:
  - justification for any support activity (events and operational requirements)
  - preventive and corrective maintenance activities
  - operational support activities
  - maintenance level and maintenance location information
- field/fleet supportability improvements and status
- lessons learned and status
- support drivers
- new technology requirements

In general, the LSA PP can be a part of an overall IPS management plan or IPS program plan. It is recommended to have a separate document in case of more complex projects. Customers and contractors must agree and harmonize many other aspects during an LSA Guidance Conference (LSA GC).

2.1.2 General requirements

The information in the LSA PP includes, but is not limited to, the following elements: Moreover, the range and depth of information for each element is tailored to the project phases.

- a description of how the LSA program will be conducted to meet the system and supportability requirements defined in the applicable project documents
- a description of management structure and authorities applicable to LSA. This includes the correlation between line, service, staff, and policy organizations.
- the identification of each LSA task and how it will be performed
- a schedule with estimated start and completion points for each LSA program activity or task. It is necessary to identify scheduled relationships with other IPS program requirements and associated system engineering activities.
- a description of how LSA activities and data will interface with other IPS and system-oriented tasks and data. This description will include analysis and data interfaces with the following programs, as applicable:
  - Product design

- Product maintainability
- Product reliability
- Product testability
- facilities and infrastructure
- Human Factors Integration (HFI)
- Initial Provisioning (IP)
- PHST
- parts control
- standardization
- support and test equipment
- survivability
- system safety
- technical documentation
- test and evaluation
- training and training equipment

– a breakdown structure identification of items, including software, which are the object of LSA performance (LSA candidates) and documentation. Identification of an LSA Candidate Item List (CIL), and criteria for LSA candidate item selection. The list must include all items recommended for analysis, items which are not recommended, and the appropriate rationale for selection or non-selection.
– an explanation of the breakdown element numbering system
– the method of dissemination of supportability and supportability-related design requirements to designers and associated personnel
– the method of dissemination of supportability and supportability-related design requirements to subcontractors and the relevant controls under such circumstances
– government data to be furnished to the contractor
– the procedures for updating and validating LSA data to include configuration control procedures for LSA data
– LSA requirements on Government Furnished Equipment/Materiel (GFE) and subcontractor/vendor furnished materiel, including end items of support equipment
– the procedures to evaluate the status and control of each task, and to identify the organizational unit with the authority and responsibility for executing each task
– the procedures, methods and controls for identifying and recording design problems or deficiencies affecting supportability, required corrective activity, and the status of actions taken to solve the problems
– a description of the data collection system used by the contractor to document, disseminate and control LSA and related design data

Note

The range and depth of information for each element is tailored to the project phases.

2.1.3 Functional requirements identification

This activity identifies the required operations and support functions for the Product. The primary inputs are the results of supportability and supportability-related design factors, as well as results of reliability, maintainability and testability programs and analyses. The results of this task form the basis for support system alternatives. LSA personnel will revise functional block diagrams as applicable, identifying subsystem component changes and their functional interaction within the subsystem and with associated subsystem components. To do so, the LSA personnel must cooperate and interact with design engineering, maintainability, testability, human engineering, and technical manuals. LSA personnel will then evaluate and identify operations and maintenance actions required to implement the maintenance and operational concept for LSA candidates.

The functional requirements for Product maintenance and operation include:

– inspection

- servicing
- testing
- operation
- repair
- replace
- configuration for specific usage

2.1.4 Unique functional requirements

The same process used to identify maintenance and operations functions will identify any unique functional requirement. These unique functional requirements are normally associated with new technologies and equipment incorporated in the Product or its support system. The process will also consider hazardous materials, hazardous waste, and environmental pollutants.

Although the same analysis techniques are used to identify and evaluate these unique functions, it is necessary to pay special attention to them because they have a tendency to exhibit higher risks. The product support organization will track configuration changes, and identify and evaluate their impact on identified unique functional requirements.

2.1.5 LSA program plan - generic example

The LSA PP is a crucial output document of the LSA GC. It contains general sections and descriptive subsections, followed by appendices. The general section must cover at least the:

- management structure of the program for both the contractor and customer
- time schedule of the program and meeting calendar
- definition of milestones
- responsibilities and reporting levels
- change process (categories and levels of authority)
- risk management
- work share agreements

Fig 1 shows a simplified and generic example of the structure of an LSA PP:

Project XXX	LSA Program Plan	Date
CONTENT		
1	General	
2	Scope	
3	General program definitions	
3.1	Management structure of the program on customer side	
3.2	Management structure of the program on contractor side	
3.3	Time schedule of the program	
3.4	Definition of milestones	
3.5	Meeting calendar	
3.6	Responsibilities and reporting levels	
APPENDICES		
Appendix A	Purpose of LSA and of supportability analysis tasks	
Appendix B	System breakdown methodology	
Appendix C	Breakdown depth	
Appendix D	Rules for LSA candidate selection	
Appendix E	Candidate Item List	
Appendix F	Rules for analysis tasks selection	
Appendix G	Rules for performance measuring and verification	
Appendix H	IT aspects	

ICN-B6865-S3000L0006-002-01

Fig 1 Example of content of an LSA PP

The LSA PP appendices will provide additional clarification and specific needs, which support the content in the subsections. It is possible to cover risk management issues in a separate appendix or include it into the rules for performance measuring and verification.

2.1.5.1 Purpose of LSA and supportability analysis activities

As an outcome of the LSA GC, the contractor and customer must agree on the principles of use for data coming from supportability analyses. Therefore, it is strongly recommended to carefully select which data elements will be documented and then link the data to their corresponding purpose. This also applies to supportability analysis activities. The selection must consider technical and economic aspects, especially for very extensive analysis activities, for example:

- a detailed or simplified LORA
- optimization methods, such as via simulation runs
- a detailed Preventive Maintenance Analysis (PMA)

2.1.5.2 Hierarchical Product breakdown methodology

This appendix to the LSA PP must detail the rules for a Product breakdown methodology. The customer and the contractor must decide how to identify breakdown elements, for example, using the S1000D Standard Numbering System (SNS), Logistics Support Analysis Control Number (LCN) in accordance with MIL-STD 1388-2B or GEIA-STD-0007, or any other identification system preferred by the customer.

In case of different Product breakdown methodologies (eg, physical and functional) within a program carried out in parallel, it is necessary to clarify whether and how to interconnect these different Product breakdowns. Additionally, it is necessary to document the purpose of its use (eg, a functional breakdown).

Note

It is recommended that detailed examples be included in the appendix, covering all possible cases (eg, how software will be incorporated within the hierarchical breakdown, how to cover different variants of a Product). In case of complex projects, a separate document can be used to determine Product breakdown structure and coding/numbering method for the corresponding Breakdown Element Identifier (BEI).

2.1.5.3 Breakdown depth for the item under analysis

In addition to the decisions required for the types of Product breakdown, the breakdown depth is a crucial attribute of the breakdown itself. The following aspects must be considered for a final decision:

- Is a full breakdown of the item required to identify all relevant spare parts, and is this breakdown type effective and applicable?
- Is a breakdown that only contains Line Replaceable Units (LRU) applicable and effective?
- What depth of breakdown is required to identify all spare parts for a repair concept at the authorized level?
- For what purpose a functional breakdown is established (eg, for PMA) and how is the functional breakdown linked to the physical breakdown?

2.1.5.4 LSA candidate selection criteria

The criteria for LSA candidate selection must be documented in an appendix of the LSA PP. The appendix can also include a decision flowchart.

2.1.5.5 Candidate item list

As an input to the LSA GC, the contractor must prepare a draft of the CIL, as an input document. During the LSA GC, the contractor and the customer must discuss this draft to produce an agreed CIL. The CIL will be a formal output document of the LSA GC.

Note

For the LSA GC, a completed CIL is often not available. The CIL is a living object during the design and development phase (and also later during in-service phase) and must be continuously updated as required (eg, by design changes). As a minimum, LSA candidate selection criteria must be determined at the LSA GC.

2.1.5.6 Potential analysis activities for candidate items

The CIL contains the selection of LSA candidates from the Product breakdown and can also contain the analysis activities for each LSA candidate. It is necessary to clarify the selection of analysis activities, as well as the depth of the analysis. For example, Maintenance Task Analysis (MTA) must determine whether the description of the maintenance task must be detailed in full or in Simplified Technical English (STE), or whether to include personnel requirements for the project.

Finally, the CIL can serve as an LSA program overview that summarizes the complete effort. Additionally, the CIL can be a helpful management tool, documenting the progress of the LSA program.

Note

It is necessary to generate a drafted CIL as a final output from the LSA GC. It is recommended that this list be designated as a reference table and part of the projects master data to be shared and synchronized across the project's master data management system. It can serve as a monitoring tool to supervise the progress of the project. This will avoid duplicating data, such as breakdown information, in external lists that are not synchronized to master data.

2.1.5.7 Performance measurement and verification of LSA activities

It is necessary to evaluate the performance of LSA activities continuously, and document it, for example, by status information. The criteria, when an analysis activity is fulfilled or when an LSA candidate must be clear and documented in a list of acceptance criteria that are part of an appendix within the LSA Guidance Document (LSA GD). The process and the rules for performance measurement and verification must be clear. Customers and contractors must observe these rules for each review/delivery of any analysis result.

2.1.5.8 IT aspects

It is recommended that the use of different software packages for the same analysis activities at different locations is avoided. Integration and harmonization processes can be extensive and are a potential risk factor. However, many programs currently use different software packages because of existing licenses, contractual constraints, or IT environment necessities. For any task to be performed, in this case, it is recommended to establish a common working environment which can be used by all LSA program stakeholders, or, as a minimum, use one common suitable software package compatible with the IT systems of all parties involved.

LSA GC must indicate the decisions about software packages. All involved parties must harmonize and agree to any change to software releases during a program.

2.1.6 LSA guidance document - generic example

The LSA GD is another crucial output document of the LSA GC. It contains the business rules of the LSA process on reporting, data element definitions, data exchange procedures, and selection of software packages to perform analysis activities.

Fig 2 provides an example of content for an LSA GD.

Project XXX	LSA Guidance Document	Date
CONTENT		
1	General	
2	Scope	
3	Implementation rules	
3.1	Simple data element list (DEL)	
3.2	IT aspects	
3.3	Product breakdown implementation rules	
3.4	Documentation of task requirements	
3.5	Documentation of MTA	
3.6	Definition of required reports	
3.7	Reporting process	
3.8	Data exchange process	
APPENDICES		
Appendix A	Detailed Data Element List (DEL), including input instructions and valid values for classifications	
Appendix B	List of selected software packages per analysis task	
Appendix C	Examples for documentation of task requirements	
Appendix D	Examples for MTA documentation	
Appendix E	Report examples	
Appendix F	Mathematical appendix	

ICN-B6865-S3000L0007-002-01

Fig 2 Example of content of an LSA GD

The LSA GD includes a general section and a number of appendices. In the general section, business rules definition covers at least:

- **Data Element List (DEL)**. This is a list of all the required data that must be recorded.
- **Reporting process and definition of required reports.** It is necessary to define and format all report content. Any data elements required for the reports must be available and accessible. All contracting parties must agree on content, format, and derived calculations. Any business rules necessary to calculate and/or select source data must be identified and documented.
- **Data exchange process**. For a proper exchange of information, the customer and the contractor must establish and harmonize a data exchange process. In programs involving several customers and contractors, a clear process for data integration and harmonization must be established. A time schedule must be agreed to during the LSA GC, with clarification on the consequences of missing the time schedule.

Note

Chap 20 provides information about recommended data exchange format (XML schema).

Note

The LSA GD can also cover additional aspects like the detailed explanation of the Product breakdown methodology or the detailed description how to document the results of the MTA (including LORA methodology and LORA decision). It is recommended to include appropriate examples.

2.1.6.1 Data element list and input instructions

The detailed DEL describes the context and description of the collected data. This includes syntax (if required) and allowed codification. In addition, a mathematical appendix can detail calculation methods of numeric values.

It is recommended that a simple draft DEL is created as an input to the LSA GC, to support the initial selection of data elements. The justification for the project's LSA DEL selection is based on the following questions:

- Is the data element required for the proof of any specification values?
- Is the data element required for the calculation of any supportability parameters?
- Is the data element required for disciplines such as technical publication, material support, identification of special support equipment, and identification of facilities or training requirements?
- Is the data element required for the performance proof of the LSA itself?
- Is the data element required to document the results of supportability analysis activities?
- Is the data element required for any report, necessary for the customer and/or the contractor?
- Is the data element a special interest for the customer/contractor (eg, internal usage)?

Note

It is recommended that data elements are selected, based on a logical, traceable need for project requirements. Any listed data element that does not relate to a requirement or the development (calculation) of a requirement is a candidate for elimination.

The DEL must be arranged during the LSA GC and will be a part of the official LSA GD. Nevertheless, it must be considered a living document. Updates to the DEL occur due to additional needs that arise during program maturation (eg, matured project design as indicated by contractor and customer).

2.1.6.2 List of selected software packages

In the LSA GC, the decision must be made, or at least prepared, as to which LSA software packages will be used by the contractor and the customer to support the required analysis activities. The selection of software packages can be contractual. It is necessary to investigate carefully any upgrade or complete change of software packages within the project, including all participants involved with such changes in the IT department.

2.1.6.3 Examples for documentation of task requirements

The LSA GD can include examples to ensure the correct documentation of different types of task requirements which justify the corresponding task. This can cover corrective and preventive maintenance task requirements, operational task requirements, and task requirements from other sources (eg, initiated by authorities, customer needs, regulations by law).

2.1.6.4 Examples for MTA documentation

The LSA GD can include examples to ensure the correct documentation of tasks. This can include the correct usage of subtasks by definition and subtasks by reference, correct assignment of warnings, cautions and notes and the correct assignment of task resources on task or subtask level, as required and appropriate.

2.1.6.5 Reports examples

The LSA GD can include examples to ensure a common understanding of reporting. Those examples can include an overview of required data elements, report formatting, sorting, filling and calculations for each relevant report provided by the LSA program.

2.1.6.6 Mathematical appendix

It is possible to document methods of calculation to complete the business rules. Both the customer and the contractor must evaluate the calculation of values, which must be available for review to the stakeholders. Mathematical appendices cover two basic aspects:

- Mathematical equations and their explanations, including a list of mathematical abbreviations and symbols used
- Sources and business rule logic from which the data is obtained for calculation

2.2 LSA program organization

The program manager is the final authority for the implementation of the LSA program integrated into the Product development organization. LSA is part of the system engineering integration. LSA is an integral element in the IPT commitment to provide the highest quality Products and services at the lowest possible cost.

The system engineering integration organization is responsible for defining the requirements analysis for the LSA program, which are to:

- perform the functional analysis and allocation, and providing them to the IPT
- provide Product and process solutions that meet the supportability requirements
- provide system analysis and control activities through the system engineering process

IPT leaders are responsible for completing the LSA program on their Product.

The IPS manager is responsible for monitoring LSA activities performed by all teams, ensuring consistency across the program and for delivery of LSA data submittals (if required). The program manager has final authority for all program functions. Supportability is an active member of the IPT. The objective of these teams is to develop guidance on system design to meet performance, producibility, and supportability requirements, and achieve low Life Cycle Costs (LCC).

2.3 LSA management responsibilities

An overview of the potential responsibilities of the different players in the LSA process, is given in Para 2.3.1 through Para 2.3.4.

2.3.1 The LSA manager

The typical responsibilities of LSA managers can include, but are not limited to:

- implementing established company operating policies and procedures concerning LSA
- accomplishing quality reviews of LSA data through in-process reviews and formal reviews prior to data submittal
- addressing questions and coordinating required corrective activity for the development, implementation, and modification of integrated maintenance concepts, support resources and related problems/status
- assessing subsystem design and support concepts for supportability influence/impact for their assigned subsystems
- assisting in the resolution of problems related to the acquisition of supplier/customer-furnished supportability data that are necessary to support the assigned subsystems
- assisting in the development and implementation of formal LSA review activities for their respective system, to ensure integration of maintenance concepts and compatibility with contractor and customer requirements
- conducting LSA reviews with the customer
- coordinating evaluation of design changes for LSA impacts and ensuring impacts are reported to decision makers in case design changes will have an adverse impact on ease and cost of support
- coordinating with appropriate program and functional management to ensure problems on their assigned subsystems are solved
- documenting supportability task analyses and coordinate task scheduling and planning
- maintaining technical liaison with design/supportability engineering and IPS elements
- managing the technical aspects of the LSA process, and the documentation for considerations on design and identification of supportability resources
- monitoring/coordinating the technical aspects of LSA relative to established program objectives, schedules, and directives
- participating in customer and supplier design and program reviews, and ensuring LSA is an agenda topic as appropriate

- participating in technical coordination meetings and design reviews with suppliers and/or the customer, ensuring that each review include a formal review and assessment of supportability and related design requirements
- providing assistance and information to program and functional management to achieve contractor and customer objectives
- scheduling LSA in accordance with engineering documentation release dates/support.
- tracking supportability task accomplishment and participating in problem solving
- establishing company LSA operating policies and procedures
- supporting the organization of a program statement of work, and approving man-hour estimates for all LSA activity
- supporting the assignment of personnel to programs and projects, as required for the LSA statement of work
- monitoring and assisting LSA program managers/leads in the performance of program statements of work
- establishing training requirements for LSA personnel and supportability-related tasks

2.3.2 Program LSA managers and leads

The typical responsibilities of an overall program LSA manager (eg, required within international projects with several participating companies) can include, but are not limited to:

- ensuring LSA program requirements are being met
- implementing established company operating policies and procedures
- controlling program costs and maintaining program schedule
- maintaining a regular contact with team members, subcontractors and/or vendors
- maintaining a program interface with program management and the customer
- holding technical responsibility for the accuracy of the LSA

2.3.3 Program technical staff (system engineers and analysts)

The technical staff is responsible for the proper operation of the IT systems. This includes the evaluation and provisioning of Product support data for other IPS disciplines or for management purposes, such as:

- having responsibility for the performance of data analysis
- operating data collection software and generate reports
- managing supplier and customer interfaces for data collection
- having working relationships with engineering departments, all IPS elements, suppliers, and customer

2.3.4 Supportability analysis integrated product team

This team participates in the development of an early LSA strategy, including, but not limited to:

- developing the LSA PP
- participating in program and design reviews
- coordinating the operational requirements
- coordinating the requirements for mission hardware, software, and support system standardization
- participating in developing the requirements for comparative analysis
- coordinating the requirements for technological opportunities
- coordinating the requirements for supportability and supportability-related design factors
- performing PMA and/or processing PMA results (eg, packaging of preventive maintenance tasks, refer to Chap 10)
- participating in the development of support system alternatives
- participating in the evaluation of alternatives and trade-off analyses
- coordinating inputs from maintainability and packaging, and develop maintenance task analysis
- coordinating the requirements for early fielding analysis

- coordinating the requirements for post-production support analysis
- coordinating the requirements for supportability test, evaluation, and verification

3 Product development organization

The Product development organization integrates the unique capabilities of each engineering discipline through the Product development process. This process is a systematic approach to the integrated, concurrent design of Products and their related processes, including production and support. This approach aims to induce the developer to consider all elements of the product life cycle, from conception through disposal, including cost, schedule, performance, supportability, quality and user requirements as soon as possible.

Integrating product development during design will help achieve first-time quality. Moreover, improved compliance with requirements will:

- support completing the program within target cost
- meet affordability requirements through design for producibility and concurrent manufacturing process improvements
- reduce operational and support costs through design-to-supportability, with emphasis on reliability, testability, maintainability and life cycle cost

Fundamental characteristics of an effective integrated product development organization are:

- IPT
- clear product definition and interfaces
- multidisciplinary teams (integrated teams)
- disciplined processes
- single point accountability
- effective communications
- empowerment
- performance metrics

3.1 Integrated product teams

A product team owns each of the product items in the entire system, including supplier and customer furnished items. Resource allocation and integration normally occur through a WBS hierarchy of product teams, starting with the overall system team. The workflow runs downward from team to team, to the detailed subassemblies or components. These teams represent significant, identifiable subsystems, sub-subsystems and components, operation planning and support segments.

Each level in the organization is fully accountable for the aggregate of its subordinate teams, as well as for the integration and overall performance of its Product. At all levels, team leaders assume full responsibility, authority, and accountability of their teams. A single, unbroken line of authority runs through the team leaders, from the program manager to the subassembly or component team leaders. Team leaders allocate roles and responsibilities based on specific, tangible support products. Each team member has a clear charter that focuses on the development or creation of a support product.

3.2 Multidisciplinary teams

Multidisciplinary teams ensure identification of design conflicts by taking into consideration Product life cycle as early as possible. Multidisciplinary teams include personnel with different perspectives and a common mission. Team members share a common effort, whether they are skilled in:

- design
- quality assurance
- manufacturing
- reliability, testability and maintainability

- safety
- technical data and technical publication
- support equipment
- supply support
- facilities
- technical services
- PHST
- personnel and training

The objective of an integrated team is to identify and resolve as many design conflicts as possible in the shortest time, with the understanding that each perspective is of equal value. Each team member brings a unique perspective to the creation of a single, concurrently engineered design. Without this integration of perspectives, there is no integrated support product development, nor concurrently engineered design.

3.3 Single point accountability

The IPT leader is accountable for the development of all Products. This feature is beneficial to avoiding redundant efforts and clearly defining the purpose.

The IPT leader brings together the right resources (eg, manpower, skills, budget and facilities) to achieve the goal.

3.4 Empowerment

Empowerment enables IPT members to achieve their mission and provides them with the ability to influence and control their disciplines destiny to the maximum extent possible. Team members are empowered enough to fulfill their individual roles and responsibilities, but their ultimate destiny is controlled by the performance of their teammates. By definition, each member within a team depends on the other team members. Empowerment means providing sufficient and adequate resources to team members to help them performing their specified roles and assuming their responsibilities.

3.5 Management plans

Corrective actions for resource problems consist of procurement actions (eg, early orders for long-lead-time items or changes to schedule, cost, and budget), and support concept changes or design changes. Technical analysis reviews and schedules will bring these changes in resources to the attention of the IPS management. The responsible functional management will work on the identified problems. The responsible subsystem supportability manager will ensure that appropriate action is assigned and that proposed changes are implemented.

3.6 Clear product definition and interfaces

Mission success depends on a clear and complete understanding of the Product that the team is accountable for providing. In addition to requirements like performance, supportability, and producibility, the Product description indicates how the Product must relate or interact with other Products.

3.7 Disciplined processes

The processes for designing, developing, testing, producing and supporting system and support system Products are founded upon accepted standardized principles. The focus of integrated Product development is a rigorous and consistent application of standardized processes for the development of all Products, along with a continuous search for methods to improve the process. As the processes are documented, the team is committed to management by process rather than management of the Product.

3.8 Effective communications

In the integrated Product development process, the term communication means that a free and open exchange of data, information and opinions is the fuel that keeps the process working.

Effective communication establishes the roles and responsibilities of the teams, defines the team's products, establishes team goals and objectives, and favors design conflict identification and resolution. It is made possible by having:

- a collocation of team members
- regularly scheduled IPT meetings
- all-hands meetings
- functional and program staff meetings
- effective use of memos
- use of electronic communications (eg, emails, video conferences)
- an exchange of up-to-date data and information

3.9 Performance metrics

It is important to know how well each IPT is progressing toward development of their Products. To this end, it is necessary to identify critical parameters in the product development process and establish an associated metric. The parameters selected by an individual or team must define:

- product quality
- schedule
- cost
- risk assessment

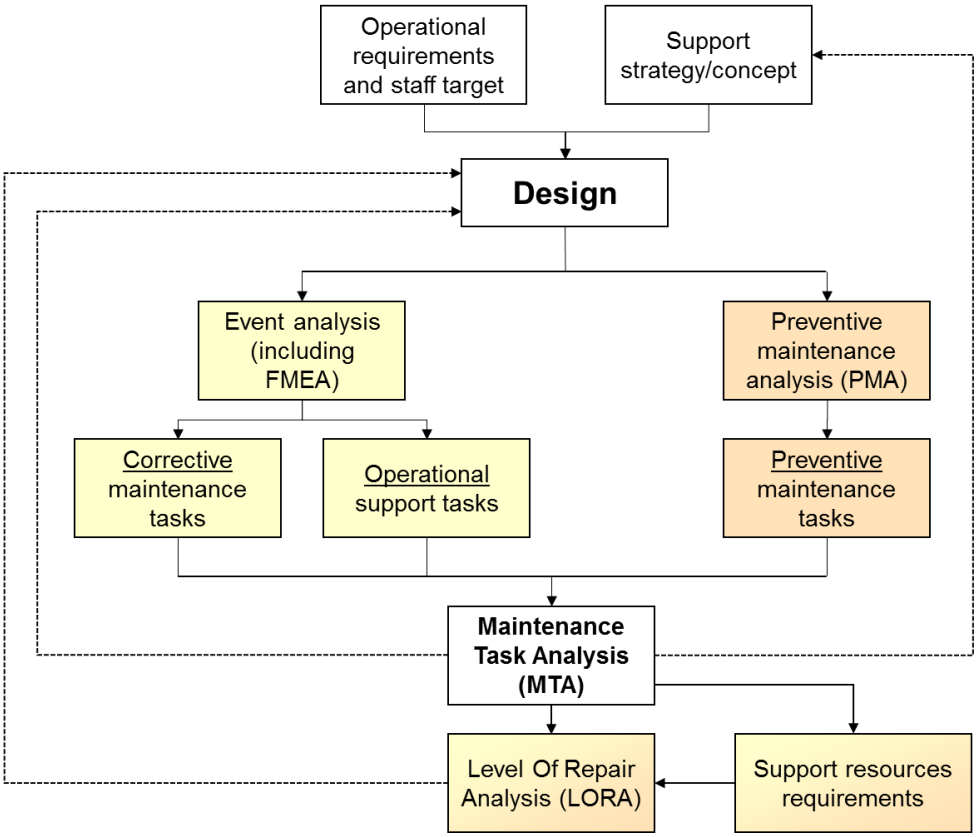
3.10 Implementation

Design reviews and technical information meetings take place throughout the program. Host design reviews and in-house design reviews ensure that design program goals and objectives are being achieved. LSA analysts and IPS personnel will be aware of supportability and supportability-related design issues by taking part to design reviews meetings and/or by receiving the relevant minutes of meeting. Supportability concerns involving equipment design that arise during meetings, or throughout the overall and continuous design process, are brought to the attention of the integrated Product engineers. It is necessary to document all information on design-related supportability issues, including status, to ensure a timely and appropriate resolution. LSA reviews are conducted in conjunction with scheduled design reviews.

3.10.1 LSA activities and performance

The LSA process (refer to Fig 3) applies selected quantitative methods to:

- determine and establish, at the beginning of the process, supportability criteria as an input to system design
- evaluate various design alternatives
- identify and provide support resources
- provide feedback to design, if required, based on the results of MTA and LORA
- assess, at the end of the process, the system support capability during use



ICN-B6865-S3000L0095-004-01

Fig 3 Simplified LSA process

The LSA program is based on an integrated product definition. The following major elements can determine the LSA program:

- an LSA program plan that identifies all the required LSA activities to influence design for supportability and determines the appropriate resources
- scheduling that identifies the timing of LSA requirements. LSA schedules are based on the needs of each project phase, and are established to support and be mutually beneficial to other project requirements.
- assignment of LSA activities to design, supportability, and IPS personnel with suitable skills and qualification
- effective management of a wide range of design, supportability, and IPS disciplines

3.10.2 Develop an early LSA strategy

It is necessary to develop an LSA strategy to be performed early in the acquisition program. This strategy will identify the scope of the proposed supportability objectives for the Product/equipment, and the qualitative analysis to provide the best cost-benefit ratio for the acquisition program. The LSA requirements will be analyzed in order to establish a comprehensive LSA program, including quantitative, qualitative, and test LSA requirements for Product systems, subsystems, and components. The result is a tailored, cost-effective program that performs any procedure required by the customer, and provides a system that meets or exceeds any customer requirement on LSA.

3.11 Program management principles

The product support team is responsible for developing the support system. This team is a subset of the IPT responsible for the design, development, manufacture, operation, safety, and support of a specific Product.

3.11.1 Integrated product support management

It is necessary to collaborate constantly with the IPS manager to provide continuity between program management direction and the development of the LSA results. Maintaining a proper interface between the LSA manager and other functional managers ensures a smooth execution of the LSA program.

The IPS manager is responsible for providing the program teams with IPS and supportability expertise and resources. In the Product development process, each IPT leader controls the budget and is accountable for the development of Products by that team. Consequently, most supportability personnel are assigned to and work with the different IPT. Supportability resources are provided to the different IPT to:

- incorporate supportability into the design of the Product and support system by providing reliability, maintainability, testability, human engineering, integrated diagnostics, and environmental suitability resources
- develop the support and training systems, and plan supportability resources by providing supply support, support equipment, technical data, facilities, PHST, manpower and personnel, as well as training and training equipment personnel
- accomplish supportability engineering in systems engineering and design process for Product development
- provide for LSA, standardization, interchangeability and interoperability, and environmental assurance

3.11.2 Operating product support organizational structures

It is possible to align the operating organizational structure for a support program with the WBS. The hierarchical characteristics of the WBS can be used to establish lines of responsibility, authority, accountability, and reporting chains from the lowest level of responsibility up to the program manager. Integration of supportability into the organization aims at:

- reflecting each supportability element identified by the WBS and system specification
- providing effective interaction for supportability-related design functions that are crucial to developing the support system, and to achieving readiness goals

Each IPT manager bears total responsibility for all aspects concerning the team's support products.

At each organizational level, Product team leaders combine the responsibilities of all subordinate Product teams. As leaders of the IPT and responsible for all Products, Product managers lead the responsibility and reporting chain, as they bear responsibility for all products. A significant feature of this type of organization is that every Product has a specific point of responsibility, authority, and accountability in the organization.

3.11.3 Formal support organizations

Functional leaders, such as leaders for Product support, production, and engineering departments, are members of the Product team and provide support to the program manager. In that capacity, these leaders bring functional expertise and resources to assist in program execution and ensure uniform, timely assignment of needed skills to the Product teams to assist in program execution and ensure uniform, timely assignment of added skill to the different IPT.

3.12 Product support management objectives and policies

3.12.1 Product support

Integrating supportability into the Product development program ensures:

- design reflects test data assessment, supportability alternatives, and tradeoff evaluations
- detailed specification requirements are provided
- supportability resource planning is adjusted as necessary
- operational availability and readiness thresholds are met
- the item is supportable in the expected operational environment

- the operational environments are accurately assessed
- the support system achieves expected performance

An underlying objective of the product support program is to identify and resolve as early as possible any supportability technical risk, prior to beginning Product production and deployment.

3.12.2 Product support management policies

To achieve supportability management objectives, a proper organization must be established for design-for-supportability through the integration of design and the development of the support system and training system. To this end, it is necessary to:

- structure an integrated Product development organization, which provides for active supportability participation and influence on design
- structure the IPS process to interact constantly, on a working level, with design engineering through the system engineering process
- plan to work closely with customers and suppliers to develop the Product

Furthermore, it is necessary to establish an LSA process that:

- provides analysis procedures for integrating supportability requirements into the baseline design
- requires the support system configuration to match the Product design configuration
- provides the detailed maintenance planning and bottom-up identification of total supportability resource requirements

To provide further control, it is necessary to establish a reporting system for IPS program progress, status and management. The system will document that program design, development, test and evaluation, and transition accomplishments meet or exceed supportability priorities and developing supportability requirements.

4 Associated parts of the S3000L data model

The documentation of the associated data to this chapter is supported by the following Units of Functionality (UoF), refer to Chap 19:

- S3000L UoF Product and Project
-

Chapter 3

LSA process

Table of contents

List of tables

List of figures

References

Table 1 References

Chap No./Document No.	Title
Chap 5	Influence on design
Chap 7	Corrective maintenance analysis
Chap 8	Damage and special event analysis
Chap 9	Operational support analysis
Chap 10	Development of a scheduled maintenance program
Chap 12	Maintenance task analysis
Chap 13	Software support analysis

Chap No./Document No.	Title
Chap 14	Life cycle cost considerations
Chap 15	Obsolescence analysis
Chap 16	Disposal analysis
Chap 17	In-Service LSA
Chap 19	Data model
Chap 20	Data exchange
S1000D	International specification for technical publications using a common source database
S4000P	International specification for developing and continuously improving preventive maintenance
MIL-STD 1388-2B	DoD requirements for a Logistics Support Analysis Record
SAE ARP5580	Recommended Failure Modes and Effects Analysis (FMEA) Practices for Non-Automobile Applications

1 General

1.1 Introduction

When introducing a new Product, it is necessary to make available all supportability requirements in a timely manner. This requires projects to establish a process that ensures the Product design takes into account supportability requirements. This process includes several analysis activities concerning a wide range of support considerations. The customer and the contractor must agree on the activities necessary to achieve proper supportability. Early consideration of supportability aspects is increasingly important with respect to both operational and economic aspects. Customers cannot accept a product that is not cost-effective to operate and maintain.

Modern products very often contain software elements. The overall Logistics Support Analysis (LSA) process for software and hardware is similar. Therefore, this chapter is valid for the supportability of software, as well as hardware. However, Chap 13 contains additional aspects specific to software that it is necessary to consider.

Product support activities are significant in terms of cost. Support costs are much higher than acquisition costs in the life cycle of a complex technical Product. Because of this, there is a trend in projects to consider supportability aspects as important as performance aspects.

Note

What is the meaning of the term logistics? There is certainly a tendency to associate this term to transportation. However, the term logistics is also used when referring to Product support activities. In the past, logistics was often used in the context of supportability. Logistics support covers the supportability characteristics with respect to required maintenance (both, corrective and preventive) and operational support, which includes for example servicing activities and Packaging, Handling, Storage and Transportation (PHST). However, the association of the term logistics to transportation can lead to misunderstandings on the actual meaning of LSA. For this reason, the tendency is to replace the term logistics by supportability in the current version of S3000L, wherever appropriate.

The word logistics is included in the acronyms Logistics Support Analysis (LSA) and Integrated Logistics Support (ILS). As this is a well-established term, Logistics Support Analysis (LSA) will still be in use in this specification. However, in order to harmonize with the rest of the ASD/AIA S-Series IPS specifications the term Integrated Product Support (IPS) will globally replace Integrated Logistics Support (ILS). Eventually, Product Support Analysis (PSA) will globally replace LSA, but not within this issue.

1.2 Purpose

This chapter is a guideline for the establishment of an effective LSA process. It considers each phase of a Product life cycle and emphasizes the importance of supportability requirements. Additionally, it describes the interaction between the contractor and the customer during the different phases of the Product's life cycle. The LSA process has two main purposes:

- additional influence on design (beside the support engineering analysis activities) to ensure appropriate supportability with the help of different analysis methods
- coordination of the creation of IPS products

The LSA process identifies requirements for spare parts, consumables, technical publication, support equipment, personnel, training, facilities, infrastructure, and software support. It supports and coordinates the development of the IPS elements. Altogether, this is a complex and extensive challenge, which requires accuracy. An efficient plan for a successful introduction of new complex technical Products calls for a properly established LSA process that the customer and the contractor must adopt and use.

1.3 Scope

The target readers of this chapter are IPS and LSA managers either working for the customer or the contractor. It aims to highlight that LSA is a powerful methodology to build the core needs for IPS realization. Additionally, the use of relevant LSA information helps performing monitoring and control functions. The application of an LSA process can positively influence the quality and commonality of IPS products. Taking into consideration the various analysis results ensures the fulfilment of the customer's needs for supportability, operability, and availability.

2 LSA process overview

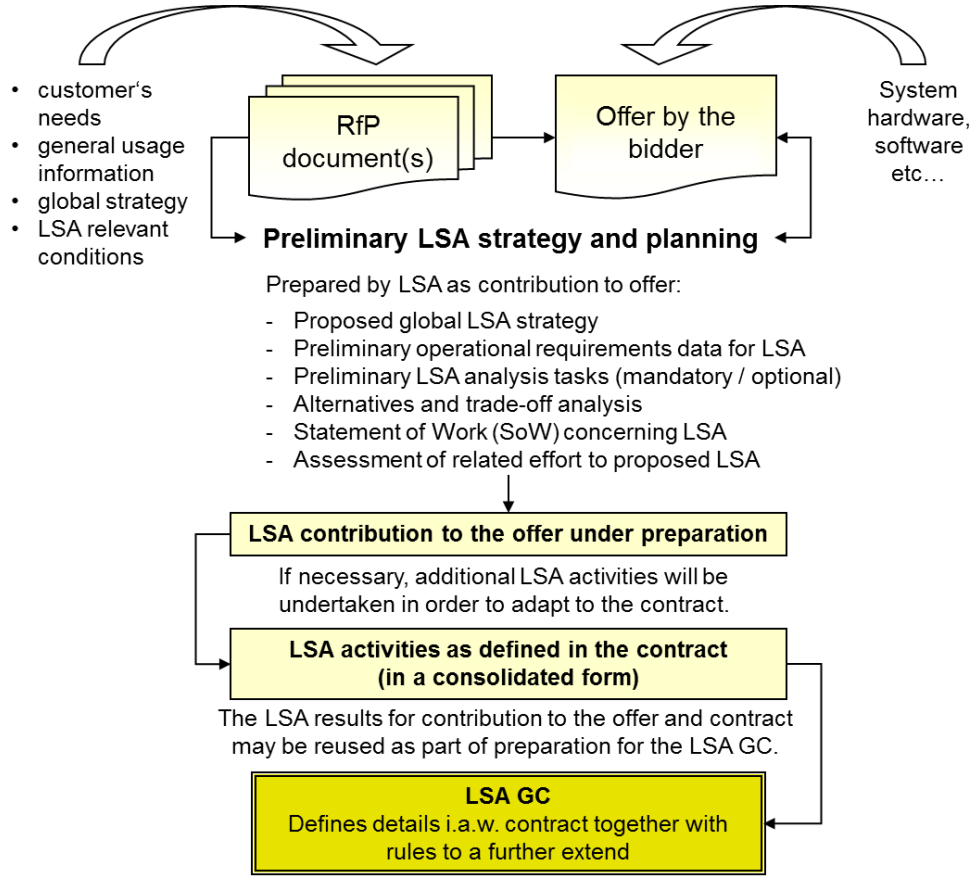
The LSA process includes a set of activities:

- activities before contract signing - Refer to Para 2.1
- activities to establish the LSA process - determination of basic requirements and preparation and attendance of the LSA Guidance Conference (LSA GC) - Refer to Para 2.2.1
- determination of scope of LSA activities by selection of LSA candidates and corresponding analysis activities per LSA candidate - Refer to Para 0
- execution of LSA process - performance of analysis activities and documentation of analysis results - Refer to Para 2.2.3

2.1 LSA activities before contract signing

It is necessary to negotiate essential decisions influencing the LSA effort before the LSA GC takes place. The LSA process usually begins when preparing an offer. The final version contained in the corresponding contract will be like the original one. This applies to any LSA significant aspect within the contract (eg, related Statement of Work (SoW)), as well as to contractual details such as deliverable items, indispensable specified values, or major milestones. For this reason, it is necessary to carry out a series of investigations before submitting the contractual offer (eg, identification of LSA activities considered mandatory, recommended, or voluntary, depending on early strategy judgment and/or the kind of Product and equipment under assessment). Refer to Fig 1.

The LSA GC serves as a means to provide the customer with the details of all activities, along with the associated rules, time schedule and deliverables (including required formats) based on the contractual agreements. Finally, it will be possible to clarify all customer questions regarding the LSA effort. LSA GC must contemplate a certain amount of project-specific adaptation, without the need to discuss contract or cost changes. Considering the iterative nature of LSA activities, a project must allow for customizations (tailoring), in order to be flexible.

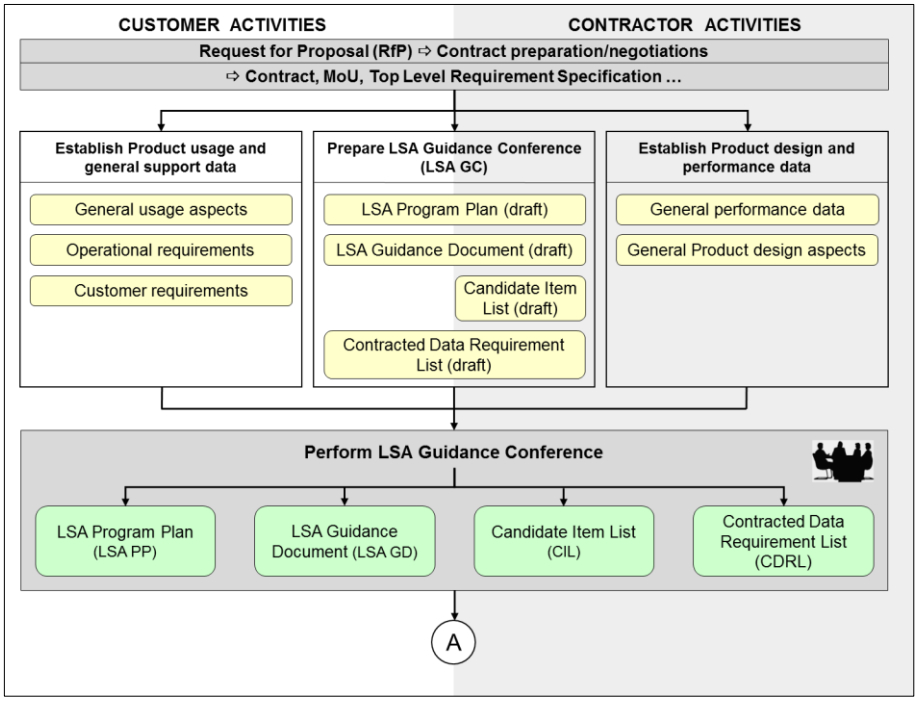


ICN-B6865-S3000L0003-002-01

Fig 1 LSA activities in conjunction with the preparation of an offer/contract

2.2 LSA process flowcharts after contract signing

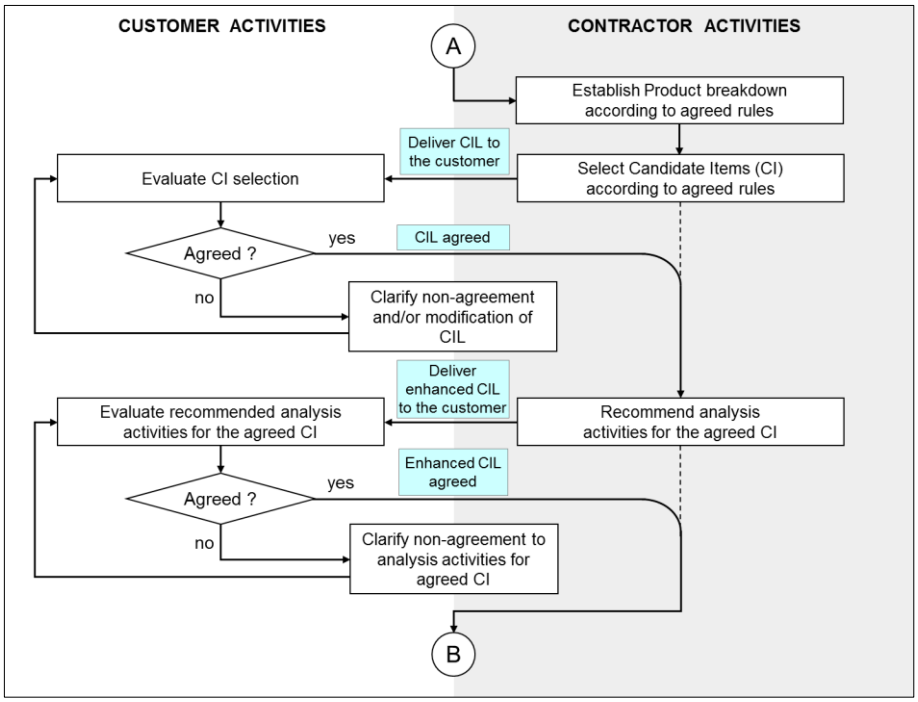
2.2.1 Establishment of LSA process



ICN-B6865-S3000L0004-003-01

Fig 2 Flowchart of LSA process at a glance (sheet 1 of 3)

2.2.2 Determination of scope of LSA activities

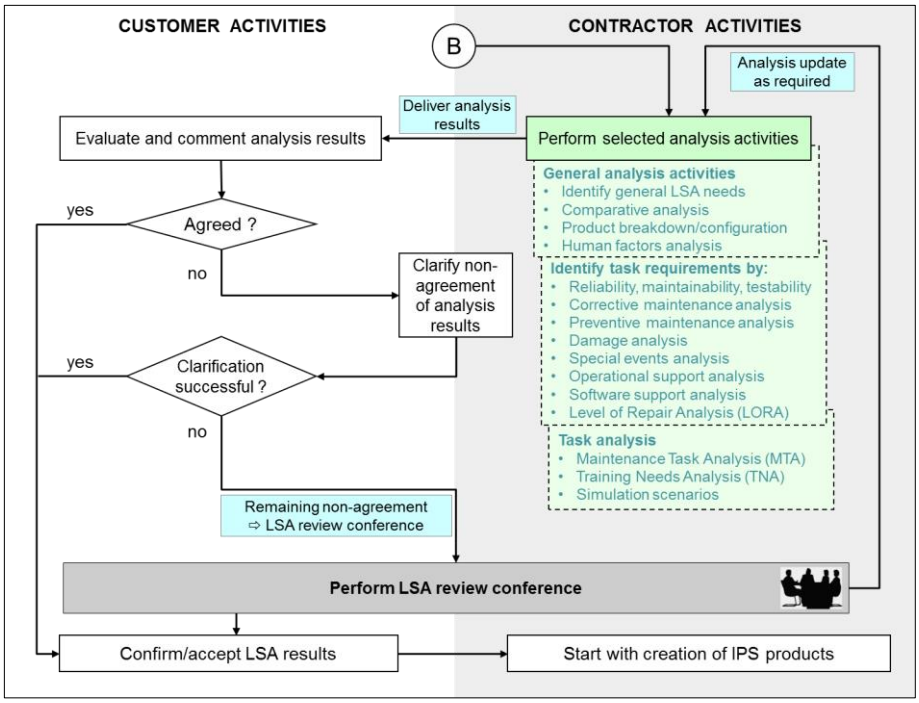


ICN-B6865-S3000L0004-003-01

Fig 3 Flowchart of LSA process at a glance (sheet 2 of 3)

The selection of LSA candidates and corresponding analysis activities is an ongoing process. After a first determination based on the results of the LSA GC, the progress of the design and development process can justify an adjustment of the CIL and analysis activities per LSA candidate item.

2.2.3 Execution of LSA process and related analysis activities



ICN-B6865-S3000L0004-003-01

Fig 4 Flowchart of LSA process at a glance (sheet 3 of 3)

The analysis activities are an ongoing process. The analysis processes lead to a continuous growth of data/information concerning the expected maintenance and operational support solution. It is necessary to harmonize all analysis results with the customer. The customer must agree to the proposals concerning all maintenance and operational support tasks. In case of any disagreement, an LSA review (refer to Para 9) can serve as an instrument to discuss and finally clarify the unresolved issues, which could not be clarified in advance. After an LSA review, the parties can come to a final agreement after clarification of the pending issues. If it is not the case, the contractor must update/repeat the analysis, taking into account the customer's objections.

2.3 LSA process details

Para 3 to Para 10 describe in detail the activities illustrated in Fig 2, Fig 3 and Fig 4.

– details concerning Para 2.2.1 - Establishment of LSA process

  - establishing Product usage and general support data, refer to Para 3
  - establishment of Product design and performance data, refer to Para 4
  - LSA GC, refer to Para 5

– details concerning Para 0 - Determination of scope of LSA activities

  - candidate item selection and identification, refer to Para 6
  - analysis activities in the context of an LSA process, refer to Para 7

- details concerning Para 2.2.3 - Execution of LSA process and related analysis activities
  - analysis workflow processes, refer to Para 7.6
  - customer involvement, refer to Para 8
  - LSA review, refer to Para 9
  - starting point and management of IPS products creation, refer to Para 10

3 Establishing Product use and general support data

To identify the pertinent supportability aspects of a new Product, it is necessary to collect all relevant information related to the intended use and document it in a set of consistently structured, mandatory documents. Sometimes, the complete relevant information is not available during the preparation phase of a project. In this case, iterative steps are necessary to provide a complete definition of the customer's use of the Product under analysis. In any case, changes to the use scenario play a crucial role in Product support and supportability analysts must consider them accordingly.

3.1 General use and support aspects

Some general decisions are necessary to establish an initial overview that identifies pertinent supportability aspects. These first decisions must consider the overall preconditions for the use of the new Product to be introduced, as well as consider some strategic aspects coming from the Product design and performance data, which are described in Para 4. A general project document describing the overall support strategy must define these general decisions. This strategy must serve as a basic guideline for the further execution of any supportability analysis and the creation of the Operational Requirements Document (ORD) and the Customer Requirements Document (CRD).

It is necessary to answer the following general questions before, or at least in parallel to the creation of the ORD and CRD:

- How will the Product be used?
  It must include a short description of the Product including key features, key requirements, and basic technical data.
- Will contractor support be required? If yes, at which maintenance levels?
  It is possible to answer this crucial question only by a deeper examination of some additional aspects described in the following questions.
- Where will the Product be operated and maintained?
  - In which operational environment will the Product be operated and maintained (from a fixed/industrial/benign environment to a mobile/austere/hostile one)?
  - Will the environment influence the item’s characteristics (eg, reliability, maintainability)?
  - Will the environment significantly influence the methods for item repair? If so, there can be a better approach than contractor support.
- How long will the Product be used (predicted service life)?
  If the Product will only be on the inventory for a few years, then contractor support is preferable to a lengthy and costly organic support structure.
- How many maintenance levels are planned for product support?
  It is necessary to establish the repair strategy clearly before writing the ORD.
- What are the features and/or typical activities within each maintenance level?
  For example, in a classical two-level maintenance concept, the typical activity is replacement of equipment and returning the faulty equipment to the contractor or to the original manufacturer for repair.

- Is it possible to use directly existing maintenance capabilities and adapt them to the support concept for the new Product?
- Is it possible and effective to use existing maintenance capabilities from other products or other similar products at or nearby the operating locations?
- Is it better to involve the customer in any repair activity of the Product, or to limit the corrective maintenance to simple equipment replacement? The involvement of the customer depends on the abilities that the customer can provide.
- What is the acceptable frequency and duration of preventive maintenance (expressed in measurable terms)?
- Are there limitations for preventive maintenance (eg, because of special preconditions concerning available personnel, competence or facilities and infrastructure)?
- How much of the software is mature? How much is unique to the customer?
  In case the delivered software is not 100% bug-free, it can take several years for the software to mature. The Product support structure must also address software support of potential upgrades required by the user. However, any modifications in the software source code cannot be regarded as maintenance. They are a design change and must also follow the process of software development, refer to Chap 13.
- Is there any software/data loading and/or unloading that the customer must perform? Which concept must be established to ensure proper function of the Product after the loading of software/data at an acceptable level (eg, simulated integrity test based on a hardware-software compliance matrix in a test bench, General Purpose Test Equipment (GPTE) concept for software, required software device, and encrypting system for loading and/or unloading).
- What is the expected need for Product upgrades due to changes in technology?
  This question determines how the customer's support structure can keep up with the changes in the Product, and how to modify the support strategy, if required. If it is difficult or even impossible for the customer to perform this, it is preferable to use contractor Product support.

It is necessary to provide the most complete answer to all these general questions. Additional questions can arise, depending on the project and on the Products included in the project. The basic principle is to use the best information available. It is recommended that the customer and the contractor create and negotiate a basic document with all general information. Ideally, this occurs before the preparation of the ORD and CRD.

3.2 Operational requirements document

The definition of operational requirements must be quantitative and qualitative. It is necessary to consider analyses conducted previously and concerning the area of operation and Product use, as these analyses identify the relations between hardware, use, and supportability. It is also necessary to define the identified operational requirements and gather metrics. Operational requirements serve as a starting point to develop supportability requirements. Each supportability requirement must be based on an operational requirement, and it is necessary to identify their relation clearly. If the basis for the supportability requirement is not clear, that requirement should be regarded with suspicion. The main goal is to identify and document the pertinent operational requirements related to the intended use of the new Product. The authors of the ORD are required to identify the Key Performance Indicators (KPI). Any parameters identified as KPI must be candidates for review if they have a negative impact on supportability attributes. At an early stage of the project, the ORD requires the Product developer to make difficult choices between “must have” and “nice to have” items. This information is vital for the supportability analysts to understand what they must support, regardless of costs, and what they can trade off.

3.2.1 General use scenario

In the general use scenario, it is necessary to collect all information on the environmental aspects of Product use. This information includes but is not limited to:

- general description of overall use areas and/or mission areas
- possible operational scenarios and requirements for each scenario
- influence on the environment by the use of the Product and resulting activities to avoid or reduce negative influence
- supportability problems arise during the life of the Product currently in use
- interaction and dependability with existing Products
- mobility requirements

3.2.2 Geographical position of locations and special conditions of each location

It is necessary to collect details on each location in which the Product will be operated and any special aspects of those locations. These details include, but are not limited to:

- number and geographical position of the operating locations
- type of each operating location
- special conditions at each operating location
- Is the location in a region currently at war?
  Threat situations require a special emergency support concept limited to the minimum amount of intervention.

- Is sufficient infrastructure available to access each operating location?
- special infrastructural requirements for reaching a location
- capabilities (existing and planned) of each location (eg, support equipment, facilities, infrastructure, personnel, supply depot, repair shop)
- interactions between different locations concerning, for example, maintenance support, from one location to another location

3.2.3 Product deployment

It is necessary to collect details about the deployment of the Product and the interactions coming from the deployment. The location of the Product affects the decision to use either integrated or contractor support. These include but are not limited to:

- number of supported Products per location
- deployment of Products per location
- interactions between the different locations concerning use

3.2.4 Use overview

Depending on the customer, defining the planned Product use provides some basic input to supportability analysis. The frequency and duration of the use, in combination with the reliability of the Product, provide the initial basis for determining the range and quantity of the required support resources. These include but are not limited to:

- key use/key missions per location
- performance parameters and constraints
  It is necessary to identify Product performance parameters such as range, accuracy, payload, or speed in measurable terms, avoiding general or potentially ambiguous terms.

- allocated operational availability and use/mission success rates
- operation per unit of time
- operation profile per operating day, week, month, or year
- use of the Product as training equipment, taking up a portion of operating time
- permanent operational conditions/possible maintenance windows
- average duration of each unique use event

- key measurement base of use per unit of time

For a detailed checklist on how to create an ORD, refer to Para 11.1.

3.3 Customer requirements document

The customer must consider all Product support requirements to ensure proper Product use. The supportability analysts can influence the supportability and design of a new Product in the most efficient way during the early development stages of the Product. One method to convey these needs is to document them in a CRD. Para 3.3.1 thru Para 3.3.9 provide the scope of the CRD.

3.3.1 Supply concept

The supply concept is crucial for logisticians. A general decision on how to organize the provision of spare parts and consumables can have high impact on the maintenance scenario. The analyst must decide whether there is a need to plan facilities and infrastructure to organize supply storage areas, and determine who will manage the supply chain when outsourcing the supply chain management. Costs (especially facility construction) and obsolescence are important aspects.

3.3.2 Support equipment concept

To be cost effective, it is advisable to acquire common support equipment, instead of special support equipment wherever possible. To reduce costs, it is important to evaluate whether it is possible to use or adapt for use the existing support equipment.

3.3.3 Personnel integration and staff training

Manpower issues are crucial to the supportability of many Products. It is necessary to plan the timely training of all support personnel. It is necessary to address acceptable risk levels, training level needs, and manpower ratios as supportability requirements. There are two types of training needs: initial and on-going. Both are important to ensure adequate personnel competence. Preserving a specific competence is often a crucial issue, considering a high level of turnover in personnel. There can be a rapid turnover in repair and maintenance personnel as well. Support planning must deal with these issues.

3.3.4 Facilities and infrastructure

Early planning for facilities and infrastructure is necessary because of the long lead times associated with site acquisition and allocation. It is necessary to plan the facilities for maintenance activities to ensure the relevant support to process needs and activity workflow.

3.3.5 IT and communication resources

IT and communication resources need proper preparation. The preparation includes, but is not limited to:

- What constraints are necessary to provide interfaces with other services?
- What is the trade-off when X architecture provides a desirable improvement in operational availability but denies access to Y communications network used by another service?
- Which IT architecture must exist or must be developed additionally?

IT resources must include all aspects such as computer hardware, computer network components, network wiring, communication protocols, software packages, data security, and standards.

This subject also requires an understanding of future capabilities. The engineer and supportability analyst must be aware of the status of other related projects, in order to design a Product that, at the time of its deployment, will interface with the other forecasted Products. It is necessary to address how the Product interacts with the planned future communications architecture. The analyst must assess the impact of expected IT system changes to determine necessary adjustments to the support structure.

3.3.6 New organizational structures

Considerations on new organizational structure cover two aspects:

- any change to the established organizational structure at a location of the customer to support and operate the Product
- any change in the organizational structure (eg, reduction in personnel) because a new Product replaces an existing Product or because the new Product is easier to maintain

Organizational structure changes have a major impact on available support infrastructure and this impact must be accounted for the development of a new Product.

3.3.7 Schedule considerations

The supportability analyst is obviously concerned with scheduling decisions. Product support is a vital and integral part of any fielded Product. It is necessary to consider the risk of delays and/or inefficient use of resources. If supportability considerations are integrated in each project phase, the supportability schedule will be synchronized efficiently and integrated with other discipline schedules (eg, engineering or production).

3.3.8 Additional aspects

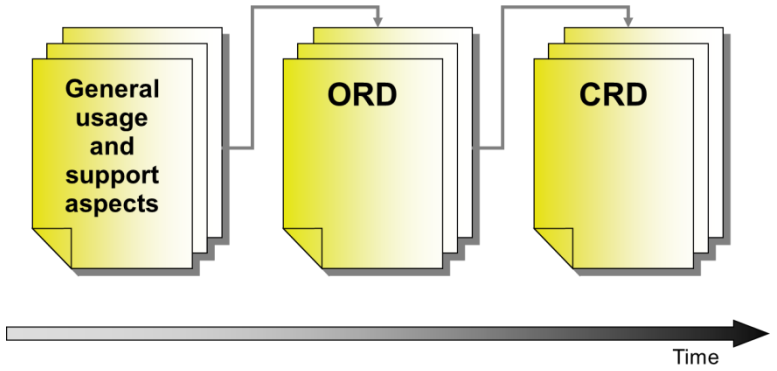
This paragraph addresses special aspects such as PHST considerations, and defines unique data requirements. Logisticians must know how and when they will use the data they require, and they must be able to separate essential data from data that can be useful to cover possible contingencies. PHST, disposal, and environmental impact considerations are far from being the main concern of system designers, developers, and users, but they are important and can potentially cause expensive consequences. Supportability analysts must understand the potential impact of these issues on the Product from the outset and must raise these issues whenever they have an impact on project planning.

3.3.9 Detailed checklist for the creation of a customer requirements document

For a detailed checklist how to create a CRD, refer to Para 11.2.

3.4 Time schedule for document creation

To ensure that all required information is available for each of the three basic documents that identify Product use data, it is necessary to identify a sequence or workflow for the creation of these documents. Refer to Fig 5.



ICN-B6865-S3000L0002-002-01

Fig 5 Schedule for the creation of the basic documents

3.5 Site surveys

Site surveys of operational units and maintenance/repair workshops can provide a significant input to the operational and customer requirements as they identify existing capabilities, resources, and potential problems. Once there are sufficient details on the operational

environment for a new Product, site surveys can help determine existing operational units and repair facilities that are most likely involved in the operation and support of the new Product. It is necessary to document the results of site surveys, and include them in the ORD (eg, as an additional appendix).

3.6 Qualification requirements

If the customer requires a qualification process, the customer and the contractor must clarify, at an early stage, which aspects of maintenance and operational activities to take into account to achieve Product qualification by the customer's authorities. The customer must define the requirements for the qualification and transmit them to the contractor together with the ORD.

3.7 Certification requirements

If a certification process is necessary for Product operation in the predicted environment, special attention must be given to the collection and documentation of the maintenance and operational support activities that are a part of the certification. These activities are necessary, regardless of their cost. All aspects of maintenance and operational support activities that influence certification must have priority. In its early stages, the project must clarify which efforts to consider in order to fulfil the requirements of the certification authority. The certification authority defines the certification requirements and the customer and contractor must be familiar with them.

The customer and the contractor must ensure that the differences between the qualification process and the certification process are evident and well known to each responsible person.

4 Establishment of Product design and performance data

LSA requires the identification and documentation of relevant Product design and performance data/information. This includes, but is not limited to:

- selection criteria concerning LSA relevant data and information - Refer to Para 4.1
- influence of overall LSA strategies and principles on LSA data selection - Refer to Para 4.2
- acceptance rules concerning the verification of values - Refer to Para 4.3
- criteria and procedural aspects for verification of projected values - Refer to Para 4.4

4.1 Selection criteria concerning LSA relevant data and information

In this context, all data and information subject to verification and control within the intended LSA process must be considered relevant. Depending on the individual Product subjected to LSA, the related contract and specifications, as well as the established IPS plan, it will be necessary to provide details on the related criteria. Relevant information must be documented as requirements in order to set a goal to be verified within the established LSA process. A general approach must consider the selection criteria described in the following paragraphs.

4.1.1 LSA data and information derived from contractual documents

It is necessary to examine contractual documents carefully, as they can contain Product design and performance features relevant to LSA that documented LSA data/information can verify. Usually, these documents cover overall Product requirements or crucial KPI to establish mandatory goals and/or thresholds potentially related to contract incentives or contract penalties. Example KPI are:

- Specified maximum maintenance man hours per operating hour
  This value can serve as a benchmark concerning successful maintenance design.

- Specified maximum Mean Time To Repair (MTTR) paired with specified MTTR percentile
  These values can serve as an indication for successful design as they relate to repair within established time constraints.

- Specified maximum failures per operating hour
  This value indicates the desired reliability of the Product and, inversely, the maintenance workload.

- Specified figures concerning minimum availabilities
  These values can indicate successful design in terms of readiness for operation.

- Testability characteristics
  These values indicate how the design ensures internal means like Built-In Test Equipment (BITE) and overall test architecture monitor vital functions, and detect and localize potential malfunctions.

- Minimum operational lifetime
  This value indicates a minimum lifetime requirement. This must indicate any risk of falling below the established threshold.

4.1.2 LSA data and information derived from Product use and general support data

It is possible to document relevant information as a baseline reference. In this context, refer to Para 3. Examples are:

- Annual Operating Requirements (AOR) with their measurement base
- number of operating locations to be considered
- number of Products to be operated at each location
- maintenance levels to be established at each location
- maintenance personnel available at each location (number of persons by specific competence)

4.1.3 LSA data and information derived from design and performance specifications

For verification and/or control purposes, it is possible to document LSA relevant parameters influencing the design and performance of the Product. In this context, refer to Chap 5. Examples are:

- specified Mean Time Between Failures (MTBF), together with the related measurement base indicating the minimum value
- reliability growth
- specified testability features concerning BITE
- specified maximum allowable time for replacement
- specified MTTR, paired with specified MTTR percentile

These values can indicate successful design in terms of repair within established time constraints.

4.1.4 LSA data and information relevant for Product certification and verification

It is also possible to gather LSA relevant information from other documents and/or analysis results originated by the design department, Configuration Management (CM), maintainability, safety, stress department or from the customer's documents. Examples are:

- special operation and/or repair limitations (eg, temperature ranges, anti-static protection requirements, clean air repair conditions)
- delivery plans
- validity information (eg, version applicability)
- hazardous classification of the functional failures that must be avoided by means of preventive maintenance
- Preventive Maintenance Task Requirements (PMTR) - Refer to S4000P
- storage limitations and/or requirements
- criticality classification of specific parts

4.2 Influence of overall LSA strategies and principles on LSA data selection

Prior to determining the relevant Product design and performance data, the tailoring of the LSA program must include an overall LSA strategy. The tailoring processes are dependent on the project's complexity, value to business units (contractor), and known risk factors. Afterwards, it is necessary to balance these constraints and performance requirements between the required analysis efforts, time, schedule, and allocated budget, aiming at the best cost/benefit value for the project. All stakeholders will review and evaluate these requirements, issues, and constraints during the LSA GC to achieve consensus. A corresponding document that represents the agreed business rules will record these findings, analyses and resulting agreements.

This tailoring activity is an iterative process that must take place at each program phase. The tailoring analysis for each phase must include the results and lessons learned from the previous phases.

4.2.1 Procedures and principles influencing selection of LSA data

The overall support strategy and principles can influence the selection of Product design and performance data in relation to the LSA process. Typical examples for influencing support strategy and principles are:

- Pre-determined two-level maintenance concept:
  Maintenance is mainly limited to two maintenance levels, one for item replacement, and the other for item repair. This will avoid requiring duplicate inventory at different repair facilities. If the equipment that requires removal and replacement at the operational site has a low failure rate and a high detectability rate (eg, by BITE), and if there is a responsive supply chain between the operational sites and suppliers, then the two-level maintenance concept is normally preferable because of overall cost considerations. With this concept, LSA data are limited to pre-determined Maintenance Levels (ML).

- Repair only at a certain ML:
  Repair is limited to user sites to achieve maximum autonomy, independent of cost-effectiveness. With this concept, repair option data are limited to the pre-determined ML.

- Limited repair at ML1 and/or ML2:
  The corrective maintenance activities can be limited to the replacement of an item (no in situ repair) to shorten any operational downtime. With this concept, maintenance tasks are limited to pre-determined criteria.

- Single source principle:
  In case of a major repair, the best solution is to choose the item supplier because of, for example, their related experience, and readily available personnel and equipment. With this concept, repair data are limited to the information provided by the supplier.

- Interim support concept:
  In order to acquire experience and reduce risks prior to final decisions on support concept, it is possible to establish an interim support phase. Within a temporary phase, the customer normally makes use of direct support provided by the contractor or by the equipment manufacturer. With this concept, support concept data are limited to preliminary information.

- Commercial Off-The-Shelf (COTS) concept:
  If the equipment used is already available on the market, related conditions must be accepted. However, a COTS equipment needs support as well. It is strongly recommended not to underestimate the effort needed to determine required maintenance and operational support based on the information provided by a supplier. Such information can be very

limited, or rather extensive. With this concept, data must reflect related supplier information/conditions.

4.3 Acceptance rules concerning the verification of values

Clearly defined acceptance rules must be established to avoid uncertainty during the verification process. These rules will allow accurate acceptance or rejection of the results for Product design and performance data relevant to LSA.

4.3.1 Category of measurement values

The type of requirement should be stated regarding the importance of the related values such as:

- mandatory values
- objectives
- thresholds (minimum or maximum values)

4.3.2 Identifying tolerances

For each identified value, it is necessary to express its associated tolerances such as:

- tolerances for the dedicated value expressed by the requirement for one single value
- tolerances concerning a group of similar values, for example, possible compensating of failure rates within a given area of items under analysis in order to meet the failure rate of the group instead of the single items, (eg, by following additional constraints)

4.3.3 Establishment of acceptance criteria

It is necessary to establish measurable acceptance rules for each specified value, along with its related requirement. These rules must identify the basic conditions (eg, fulfillment of minimum required values based on sufficient statistic confidence levels) that lead to the acceptance or rejection of specified values.

In addition, it is necessary to state the consequences of the established status information:

- status of the Item Under Analysis (IUA) indicating the acceptance of the documented value
- status of the IUA indicating the rejection of the documented value and the related justification
- status of the IUA indicating the conditional acceptance of the documented figure (eg, in general acceptable, but requiring minor adaptation)

4.3.4 Establishment of special rules

If special rules are established, it is mandatory to define the related conditions and values to avoid uncertainty:

- establishment of penalty regulations for failure to meet specified LSA significant data
- establishment of rewards in case of exceeding specified LSA significant data

4.4 Criteria and procedural aspects for verification of projected values

Finally, it is necessary to establish the criteria for verification of data and/or other information subject to verification.

4.4.1 Determination of measurable target values

For data element under consideration, it is required to establish and document the related projected value range by considering the original target along with its established tolerances and relevant acceptance rules, if any. It is possible to allocate the projected value range to a single item and/or a group of dependent items covered by relevant acceptance rules, to establish the related acceptable values that fulfill the requirements.

4.4.2 Verification of the individual projected values

It is necessary to make a comparison between the relevant data or other information derived from the analysis process and the associated acceptable values. The results must be reported to the analyst and/or involved management and indicate whether the actual LSA values are consistent with the projected values.

4.4.3 Verification method

The customer and contractor must agree on the projected values and the relevant verification method:

- verification by provision of analytical proof
- verification by analytical methods based on appropriate tests (eg, performed at a test rig)
- verification by demonstration on a prototype or serial versions of the Product
- verification in compliance with the rules of certification and/or demonstration programs
- verification by trial sessions (eg, performed by customer's staff)
- verification during long-term exercises (eg, during a defined maturity phase)

4.4.4 Verification for special purposes

Rules can be required when verification for special purposes is necessary in order to gain certifications, qualifications, or licenses.

4.5 Checklist for the LSA guidance conference

The checklist must contain all relevant proposals for design and supportability requirements. For example:

- Have Product design and performance data relevant to LSA been identified?
  - list of selected LSA data and information derived from contractual documents
  - list of selected LSA data and information derived from Product use data
  - list of selected LSA data and information derived from design and performance specifications
  - list of selected LSA data and information contained in other Product documents
  - list of selected LSA data and information subject to certification and verification
- Have acceptance rules on the verification of relevant values been identified?
  - Have measurable target values been identified for the selected data/information?
  - Has the category of measurement figures been indicated for the selected data/information (eg, mandatory values, goal values, thresholds)?
  - Have tolerances been defined for the selected data/information?
  - If applicable, have compensating rules been established?
  - Are projected values acceptable/agreed to by the customer?
  - Have the consequences of acceptance/agreement concerning status information been identified?
  - If applicable, have penalties and/or rewards regulations been established?
- Have the overall LSA strategies and principles, which influence the aspects above, been established?
  - Identify the preferred maintenance concept (eg, is a two-level maintenance mandatory?)
  - Has the supply chain management support concept been identified?
  - Have the repairs been focused on one maintenance level and, if so, which level?
  - Is repair at Maintenance Level 1 and/or 2 limited?
  - Has the single-source principle been determined?
  - Is an interim support concept applicable?

- Is it necessary to consider the COTS concept?

The details of the LSA GC checklist must be part of an LSA Guidance Document (LSA GD).

5 LSA guidance conference

The LSA GC must be the central event. Management staff and specialists from both the customer and the contractor must attend it. At this conference, the binding agreements for the performance of the LSA process must be established. During the LSA GC, the customer must agree on the selection of Product design and performance data subject to verification and control within the LSA process. This also applies to the relevant rules for the implementation of the LSA process.

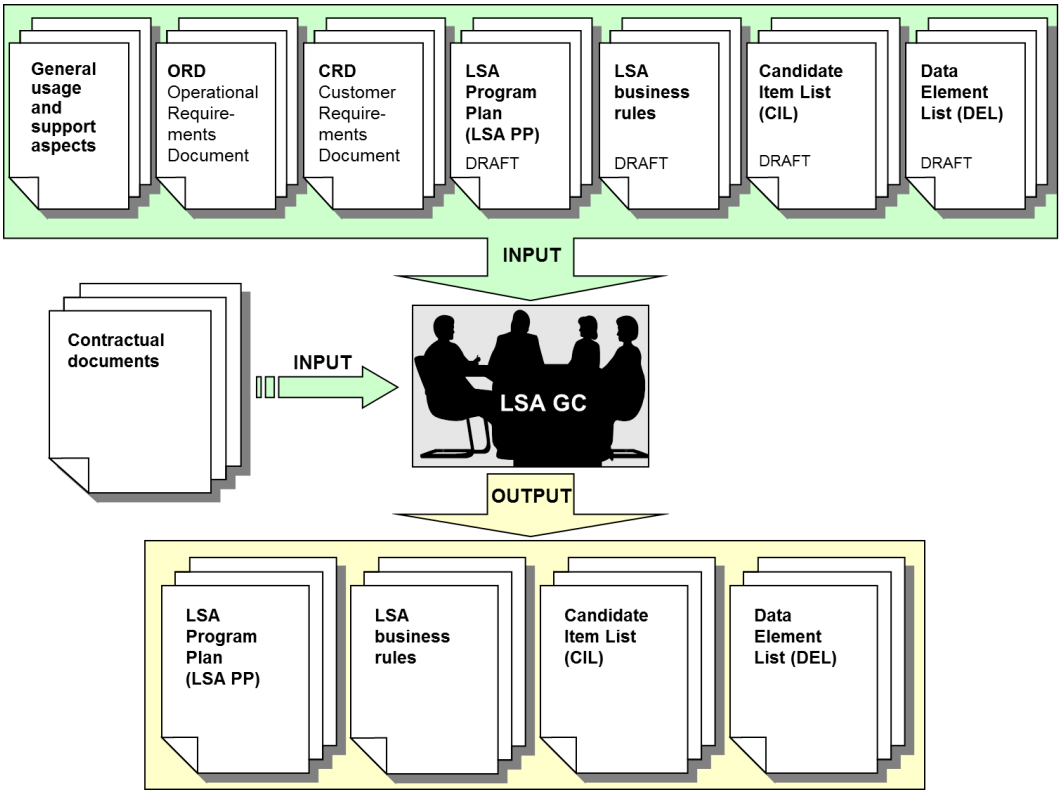
To maximize the outcome of this conference, it is necessary to prepare inputs and to have a clear understanding of the expected results and final agreements. It is recommended to have checklists concerning preparations and expectations for the LSA GC.

Note

It is recommended to avoid the use of the phrase "to be defined" as an input for any subject to be discussed during the LSA GC. Specialists must prepare concrete proposals concerning the performance of the required activities, and these proposals must serve as the basis for all discussions. Evaluate the advantages and disadvantages (cost/benefit) of the alternatives under consideration. Provide documentation of all alternative analysis and decisions.

5.1 Documents and information for LSA guidance conference

Para 5.1.1 and Para 5.1.2 include lists of applicable documents relevant for the LSA GC. These lists provide a guideline on the methodology to document the required information. The project can add additional aspects or skip some information. Fig 6 gives a summarized overview of the recommended inputs and outputs for the LSA GC.



ICN-B6865-S3000L0005-002-01

Fig 6 LSA GC, inputs and outputs

It is necessary to negotiate the majority of essential decisions influencing the LSA effort, and document them in the contract. This applies to any LSA relevant aspect within, for example, a SoW and the corresponding parts of the commercial offer, as well as for contractual details such as delivery items, indispensable specified values and major milestones. This mandates to perform a series of investigations prior to the contractual offer. Said investigations include the identification of LSA activities considered as mandatory, recommended or voluntary, depending on early strategy judgment and the Product/equipment type to be assessed. Refer to Para 2.1.

5.1.1 Input to the LSA guidance conference

Table 2 provides an overview of the required documents/information to prepare for an LSA GC.

Table 2 Required input documents for the LSA GC

Type of document	Content	Responsible party
General use and support aspects	Strategic aspects of the Product design and performance data concerning use and supportability framework. Refer to Para 4.	Customer
Contractual documents	Contractual decisions that can influence the LSA process must be available during the LSA GC and taken into account (eg, SoW, contracted LSA deliverables, milestones).	Customer and contractor

Type of document	Content	Responsible party
Operational Requirements Document (ORD)	Details about the use of the Product in the planned scenario at all operational locations. Refer to Para 3.	Customer
Customer Requirements Document (CRD)	Details of customer requirements regarding the operational scenario described in the ORD. Refer to Para 3.	Customer
LSA Program Plan (LSA PP) - DRAFT	Management plan for the LSA program	Customer and contractor
LSA business rules - DRAFT	Implementation rules for the LSA program	Customer and contractor
Candidate Item List (CIL) - DRAFT	A preliminary list of Breakdown Elements (BE) and/or parts to consider as potential LSA Candidate Items (CI). It is necessary to present at least the rules for candidate selection.	Contractor
Data element list (DEL) - DRAFT	A list of proposed data elements required to document the performance of all potential supportability analysis activities.	Contractor

5.1.2 Output of the LSA guidance conference

The result of the LSA GC must be a series of measurable rules, agreed by customer and contractor, to guide the LSA process. These rules, checklists, and documents provide a clear road map of needs and expectations between the customer and the contractor. The customer and contractor must agree to and sign this collection of official documents. These documents must at least contain the items shown in Table 3.

Table 3 List of output documents from the LSA GC

Type of document	Content	Responsible party
LSA Program Plan (LSA PP)	Initial release of the LSA PP. For details concerning an LSA PP, refer to Chap 2. Consensus of LSA GC participants required.	LSA GC participants (customer and contractor)
LSA business rules	The business rules determine the implementation of the LSA process. Typically, an appropriate document, for example an LSA GD, contain these rules. For details concerning an LSA GD, refer to Chap 2.	LSA GC participants (customer and contractor)
Candidate Item List (CIL)	List of LSA candidates containing all BE and parts selected for supportability analysis activities, including the selected analysis activities for each CI.	LSA GC participants (customer and contractor)

Type of document	Content	Responsible party
Data Element List (DEL)	A list of agreed data elements required to document the performance of all potential supportability analysis activities. The DEL can be an appendix to an LSA GD.	LSA GC participants (customer and contractor)

5.2 Candidate item list

Para 6 describes in detail the creation of the CIL, including the associated rules.

5.3 Contractual documents

Contractual conditions are the basis for any project. It is necessary to consider all contractual decisions made prior to the LSA GC. If these decisions affect the IPS process, they will also have a major impact on the selection of the supportability analysis activities.

Another contractual consideration is the significance of the output documents of the LSA GC. The supporting LSA GC documentation must support contractual requirements and needs for both the customer and the contractor.

5.4 LSA database

The documentation of LSA data/information requires proper IT support. As a general recommendation, an effective approach is to implement and maintain an LSA database based on existing standards or specifications. Chap 19 and Chap 20 of S3000L specification provide the baseline for the development and implementation of an IT solution.

During the LSA GC, it is necessary to determine the approach for the implementation of an LSA database (or alternate solutions) within an LSA program. Some basic IT requirements must be considered, including:

- selection of IT toolset
- common data source principle to avoid time-consuming and error-prone data exchange processes (one LSA database for all stakeholders, if possible)
- processing LSA data by import and export (if required)
- LSA data evaluation by creating project specific reports
- network capacity and security requirements (crucial for international projects)
- access rights to database content and database functionality (role concept)

6 Candidate item selection and identification

It is necessary to consider the LSA process to be cost-intensive. For that reason, the selection of LSA candidates and the associated analysis activities must be carried out with reasonable care to keep a good balance between the effort for LSA and the benefits obtained from the LSA process. The LSA candidates can be different in quality, therefore the type of analysis process and the depth of the analysis must vary depending on the LSA candidate type.

The selection of the LSA candidates requires a proper Product breakdown, and IPS aspects must drive the breakdown. The configuration of the Product is of crucial importance. The first baseline analysis establishes an appropriate Product breakdown, and it is necessary to lead all further technical/supportability analyses performed during the entire LSA process. The representation of a Product by Product breakdown must include a set of aspects, summarized by the general term "configuration" of a Product:

- types of Product breakdown (eg, functional or physical)
- installation location and realization by hardware (parts) or software
- multiple installations of the same part at different locations

- alternate parts for the realization at one and the same installation location
- substitute components for equipment repair
- handling of variants of equipment/components within variants of the Product (configuration aspect)

Chap 4 provides details on how to establish an appropriate Product breakdown using LSA-relevant aspects.

6.1 Definitions and general terms

First, some definitions of terms used to describe Product components in the context of supportability are introduced.

6.1.1 Candidate and non-candidate

The LSA candidate is the driver for all LSA activities. In general, any IUA on system, subsystem, equipment, module, sub module or part level can be a potential LSA candidate. The IUA can be an element of the Product breakdown, identified by a Breakdown Element Identifier (BEI), or an element from a part list identified by a part identifier (eg, a part number). The analysis of the LSA candidate aims to determine its need for any maintenance (preventive and/or corrective) and/or operational support. The set of criteria described in Para 6.3 determine whether an IUA will be part of the CIL.

In addition to this hardware-related definition, special activities and/or descriptions can be part of a Product breakdown. For example, it is possible to document the description of standard practices for the repair of structural components within a special area/chapter of the Product breakdown. In this case, these specific non-hardware breakdown elements can be LSA candidates as well because of special events or maintenance tasks associated with them. Non-candidate items do not require a detailed supportability analysis. These items do not need to appear in a Product breakdown under a unique BEI or a part identifier. In case of a complex breakdown, all items that do not fit into any of the selecting criteria given in the LSA candidate selection checklist are potential non-candidates. Items that are only subject to standard tasks and do not require any special resource are also potential non-candidates. Consumables and bulk items (eg, screws, bolts, nuts, washers), as well as wiring or fuel lines, are typical examples of non-candidate items.

6.1.2 Maintenance relevant items

It is necessary to consider any IUA affected by any maintenance activity to be relevant for supportability analysis. Table 4 provides a definition for Maintenance Relevant Item (MRI).

Table 4 Definition of MRI

Item type	Definition
Maintenance Relevant Item (MRI)	A MRI is an item that can be repaired or replaced in case of failure or damage. These items are potential LSA candidates.

6.1.3 Structural items, structure significant items and structural details

The structure of a Product, such as the bodywork of a car, is also part of the Product breakdown. Structural items can be typical LSA candidates, which can be repaired or replaced. They can also be candidates for the Preventive Maintenance Analysis (PMA), and are therefore relevant for potential PMTR. Refer to Table 5.

Table 5 Definitions in the context of structural components of a Product

Item type	Definition
Structural Item (SI)	A SI is any part of the Product structure.
Structural Significant Item (SSI)	A structural item or assembly considered crucially important because it carries aerodynamic, ground, pressure or control loads, and whose failure can affect the structural integrity necessary to operate the Product safely and/or can become a hazard to human safety, and/or can affect law and/or environmental integrity. Refer to S4000P.
Significant Detail (SD)	A limited area of an SSI or a local spot that is part of the whole SSI. Refer to S4000P.

Note

The definitions for SSI and SD are in line with those in S4000P. In order to document an SD within the Product breakdown, it is necessary to consider that a part identifier normally cannot identify an SD, because the SD is just a specific area of an SSI. For this reason, it is recommended to identify an SD using an additional BEI one level below the SSI itself.

6.1.4 Non-hardware items

A hierarchical hardware breakdown usually documents only hardware (including software, if required). It must be possible to include non-hardware items in the Product breakdown, in case there is a need to describe any general task that cannot be assigned to a specific hardware/software or that is a standard procedure for a group of items (eg, standard repair procedures for electrical wiring). Refer to Chap 4. Logical aggregation by systems/subsystems (eg, fuel system of a vehicle, hydraulic system on an aircraft) or segmenting of a Product by zones are typical examples of non-hardware items. These non-hardware items can also become LSA candidates because there can be maintenance or operational support tasks which need to be documented against these items (eg, a system test or a zonal inspection).

6.2 Classification of LSA candidates

It is recommended that categories for LSA candidates be established. This ensures keeping the analysis effort to an adequate level for each category of LSA candidate. Table 6 shows the possible categories for LSA candidates. The customer and contractor must define and harmonize the exact meaning of each category, especially for partial candidates, as it can be necessary to define more than one partial type within a specific project.

Table 6 LSA candidate categories

Candidate category	Description
Full candidate	Provide the complete range of selected LSA information applicable to the related item.
Partial candidate	Provide a partial scale of selected LSA information applicable to the related item, (eg, only remove and install information to gain access to other items, but no repair information required because the item is a safe life item).
Candidate family	Provide LSA information focused on specific requirements for similar items. For example, for a family of items such as specific wirings, harnesses, pipes, lines. Provide information on minor items of the same type (eg, clamps or connectors). Those items can only require a summary analysis for the family.

Candidate category	Description
Standard procedures candidate	Provide a partial scale of LSA information because the LSA information of other non-hardware BEI (eg, tasks concerning standard repair procedures of structure or electrical wiring) covers all relevant information for this item.

For every LSA candidate type, it is necessary to establish a list of criteria and to develop a flowchart for the decision process to support the analysts together with this list of criteria.

6.3 Selection process and criteria list

For LSA candidate selection, it is necessary to evaluate the significance of each breakdown element and/or part with respect to LSA purposes. If the agreed selection process indicates that an item is a candidate according to the selection rules, the relevant item will become an LSA candidate.

It is necessary to adapt the LSA candidate selection process to each project. Para 6.3.1 thru Para 6.3.4 give examples for typical selection criteria. Additionally, Para 11.3 includes basic LSA candidate selection flowcharts. Depending on the project, it is possible to modify some questions in the flowcharts or to adapt the selection flowchart with additional questions.

To perform the LSA candidate selection process, it is necessary to fulfil some preconditions concerning Product breakdown and the establishment of rules.

6.3.1 Preconditions for the selection process - Product breakdown

A hierarchical Product breakdown will establish the LSA candidates from the root level (in general the Product itself) down to the required depth for the sub-module or part level. Therefore, the availability of a Product breakdown is the most important precondition to perform an LSA candidate selection. It is essential that the existing breakdown be applicable to this task by considering the following criteria:

- Is the Product breakdown available in a sufficient maturity, depth and extent? Refer to Chap 4.
- Is this a preliminary issue of the Product breakdown (eg, to respond to a request by a potential customer, or to enter into a pre-contractual agreement with a potential customer)?
- Has the customer already accepted the existing or initial Product breakdown (including methodology)?

6.3.2 Preconditions for the selection process - existing analysis results

LSA candidate selection must use each existing analysis result (eg, from analysis activities performed in advance or provided by the manufacturer). Examples are:

- existing experience for equipment already used in other products
- existing maintainability, reliability or testability information/data
- existing Failure Mode and Effects Analysis (FMEA), or Failure Mode and Effects Criticality Analysis (FMECA) results

6.3.3 Preconditions for the selection process - Candidate selection rules

As a part of the LSA GC, the contractor and the customer must establish the rules for LSA candidate selection and the extent of analysis per LSA candidate. This means that, at the beginning of an LSA GC, it is only possible to present a preliminary CIL rather than a final CIL. The criteria that can serve as a basic guideline for aspects potentially relevant to LSA candidate selection include, but are not limited to:

- the item has a significant impact on maintenance concerning the related failure/defect frequency or the awaited workload
- the item requires special resources, such as highly qualified personnel or specific training, material or support equipment
- the item is subject to preventive maintenance with a scheduled interval (eg, preventive replacement of life-limited items)
- the item is subject to other preventive maintenance procedures (eg, after special events such as lightning, overheat, hail strikes, contact with obstacles)
- the item is subject to diagnostic and/or functional test tasks (eg, complete system tests)
- the item is potentially endangered by damage due to the installation area and/or the design of another item
- the item is subject to the use of new technology
- the item contains user-loadable software (including data)

The following items are potential LSA candidates that require assessment for a global point of view or for special interests:

- The item is a potential readiness driver
- The item is a potential cost driver (eg, expensive support equipment required)
- The item is a potential maintenance driver (eg, high workload expected)
- The item is subject to explicit customer interest
- The item is subject to contractual fulfillments related to LSA

The following items are potential LSA candidates that require assessment only or mainly for standardization reasons:

- The items require only removal and installation tasks in order to gain access to an LSA candidate. These items do not require full LSA activity, but it is necessary to standardize and document the involved supportability requirements.
- Documentation of general tasks. Those tasks need to be considered mainly for registration and documentation of the associated supportability requirements. Often, the general tasks are linked to non-hardware BEI.
- Groups of items that could require a summarized supportability analysis, for example a family of non-specific wirings, harnesses, pipes, lines, minor items of the same type (eg, clamps or connectors)

It is necessary to detail all the criteria above using measurable threshold values. For example, the criteria must use a clear threshold value (eg, the MTBF) to indicate whether an item is significant for maintenance with respect to the related failure/defect frequency. The above criteria fit for all types of candidates. Para 6.3.4 describes the possible criteria for the differentiation of the candidate types.

6.3.4 Preconditions for the selection process - candidate classification selection rules

To support this part of the LSA candidate selection process, Table 7 thru Table 9 provide examples of criteria for the different candidate types. Para 11.3 provides the flowcharts for candidate item selection.

Table 7 Classification criteria for full LSA candidates

Candidate classification	Classification criteria	Required answer
Full candidate	Is the item a newly developed or a major modified item (functional equipment, not valid for structure)?	Yes
	Is the item a Line Replaceable Unit (LRU)?	Yes

Candidate classification	Classification criteria	Required answer
	Is the item a repairable item?	Yes
	Is the item a low-reliability item (is low-reliability measurable)?	Yes
	Is the item relevant for maintenance? In other words, are there any dedicated maintenance tasks planned such as repair, servicing, lubrication, or calibration (no general tasks such as simple cleaning)?	Yes
	Are dedicated maintenance tasks complex, time-consuming or do they require many staff members?	Yes
	Is the required support equipment for dedicated maintenance tasks non-standard or non-existent?	Yes
	Is specific preventive maintenance, as identified by PMA, required for the item?	Yes

If the answer is "No" to all the questions in Table 7, the IUA will not be a full candidate. If the answer is "Yes" to any of the questions above, then this item is a potential full candidate and can be classified as such in the CIL.

Table 8 Classification criteria for partial LSA candidates

Candidate classification	Classification criteria	Required answer
Partial candidate	Is it necessary to remove the item to gain access?	Yes
	Is removal for access frequent for the IUA?	Yes
	Is the item a system or a subsystem that has not been previously defined as a full candidate, and will a fault location and/or a test procedure or other general maintenance procedures be described?	Yes
	Is the item a non-hardware item, for which general activities (eg, cleaning, storing, parking, mooring, general inspections) will be described?	Yes

If the answer is "No" to all questions in Table 8, the IUA will not be a partial candidate. If the answer is "Yes" to any of the questions above, then this item is a potential partial candidate and can be classified as such in the CIL.

Table 9 Classification criteria for LSA candidate family

Candidate classification	Classification criteria	Required answer
Candidate family	Is the IUA installed within the Product many times in the same or in a similar way?	Yes
	Is it possible to combine all equal or similar items into one family with common maintenance or operational support activities?	Yes

It is recommended that the LSA candidate family concept be used when a large number of equal or similar items is installed within the Product. Refer to Table 9. For example, all electrical wiring with the same properties and identical or similar connectors can be summarized under one LSA candidate family. For this family, all information relevant to maintenance (eg, a connector repair concept) is valid for all single cables of the whole family.

6.4 Influencing factors

It is necessary to take into account the following factors to select candidate items and/or establish the related rules:

- project phase to which the LSA program belongs
- complexity of the Product to be analyzed
- price of the Product to be analyzed
- budget available for the overall Product support and budget planned specifically for the LSA process. Budget limitation will always influence the number of LSA candidates that can be analyzed. In these cases, a reduction of the real maintenance and cost drivers is required.
- importance of the LSA results for the internal IPS elements
- importance of the information required by the customer and the industry management to make decisions and fulfill the contractual requirements of the LSA-related items
- items already in use under comparable conditions
    - items already in use, but not under comparable conditions
    - COTS items usable with or without any major modification
    - items already available but requiring major modification
    - newly developed items
- information expected from the LSA process (to be harmonized with/agreed by the customer during the LSA GC)
- identification of possible alternatives (for hardware, software, support concepts) and related consequences (eg, support concept, Product support requirements)
- proposal of recommended support concept
- identification of tasks associated with the intended support concept
- identification of related Product support requirements
- estimation of related Product Support Costs (PSC) as part of Life Cycle Costs (LCC)

6.5 Recommendations for LSA candidate selection

Table 10 provides a list of recommendations concerning essential and extra criteria to provide the analyst with an overview on how to handle the LSA candidate selection criteria in detail. Additionally, the table provides assistance in the selection process and recommendations to determine whether IUA selection as an LSA candidate is mandatory, recommended, or voluntary.

Table 10 Recommendations for LSA candidate selection

Aspects	Description of criteria	Consideration as potential candidate item?
Risk and criticality	Is it necessary to integrate LSA candidates from separate LSA programs? External LSA information to be integrated into one LSA program without any change (eg, LSA data for an engine derived from a separate contractor), or external LSA information to be integrated with adaptation to an LSA program that requires changes to be harmonized with the original contractor.	Mandatory
	The IUA is subject to PMA (eg, S4000P) and PMTR are identified.	Mandatory
	The structural IUA is subject to PMA and PMTR are identified.	Mandatory
	The IUA is subject to life limits.	Mandatory
	The IUA is subject to preventive maintenance, other than scheduled, or to special procedures (eg, after special events such as lightning, bird, hail strikes or contact with obstacles).	Voluntary
	The IUA is subject to the use of new technologies.	Mandatory
	The IUA is a potential cost driver (high-value items, cost limits must be defined with the customer).	Mandatory
	The IUA is a potential readiness driver because of long repair times.	Mandatory
	The IUA is a potential maintenance driver (high workload)	Mandatory
	The customer has a vested interest in the IUA (information values must be defined with the customer)	Mandatory
	The IUA is subject to LSA-related contractual fulfillment	Mandatory
Item type	Item already in use under comparable conditions	Recommended
	Item already in use, but not under comparable conditions	Mandatory
	Item already in use, but requiring major modification	Recommended
	Item newly developed	Mandatory
	COTS items	Recommended
	Part of an item family significant for maintenance	Voluntary
	Item containing user-loadable software and/or data	Mandatory
General attributes	Item is line-replaceable	Mandatory
	Item is shop-replaceable	Recommended
	Item is line-repairable	Recommended

Aspects	Description of criteria	Consideration as potential candidate item?
Maintenance significance	Item is shop-repairable	Recommended
	Item is repairable at customer depot or at industry level	Voluntary
	FMEA/FMECA available for the related item	Mandatory
	Item is potentially subject to LSA-relevant servicing	Voluntary
	Item is potentially subject to (diagnostic/functional) test	Recommended
	Item is potentially endangered by damage	Voluntary
	Item is potentially subject to general tasks	Voluntary
	Item is potentially subject to standard procedures	Voluntary
	Item is subject only to gain/undo access to LSA candidates	Voluntary

6.6 Candidate item list (draft)

For LSA candidate selection purposes, it is recommended to use a matrix to structure and document the selection results. The contractor must prepare a recommendation which reflects the proposed selection criteria and must include a first draft of the matrix. Moreover, the contractor must prepare the draft of the CIL for the LSA GC. After harmonizing the proposed selection rules between the contractor and the customer, it is possible to rework this draft.

6.7 Candidate item list as an output of LSA guidance conference

It is necessary to include the CIL in an obligatory list, reflecting the decisions taken by the customer. This CIL must be a living document in order to record any change occurring during the different project phases. During the LSA GC, (refer to Para 5) the improvements to the draft version of the CIL must be considered as one of the most important steps regarding contractual terms. In the CIL, it is possible to manage a collection of all LSA candidates and the corresponding workload from the selected analysis activities. To support the management of the LSA process, it is necessary to use status codes to document the progress of the LSA and documentation work. The codes must reflect the status for all the different contractual relevant analysis activities to be covered within the LSA process.

To increase effectiveness, it is recommended that the CIL or the candidate item identification process, accordingly, be integrated as a central management tool into an LSA IT solution. This approach enables linking the relevant management information to the corresponding items within the Product breakdown. Appropriate evaluations or reports easily provide an overview of the project.

The customer and contractor must discuss, harmonize and document any change to the CIL occurring for whatever reasons (eg, technical, budgetary, redesign). LSA and IPS managers for both the customer and the contractor must consider the CIL as a valid document to control the whole LSA process. The customer and the contractor must understand, at every stage of the project, who is the holder of the CIL issue that is contractually valid.

7 Analysis activities in the context of an LSA process

In the early life cycle, LSA must support cost-relevant decisions by performing adequate front-end analysis like Level of Repair Analysis (LORA). Where necessary, analysis results must influence design concerning supportability aspects from the beginning of the definition phase and throughout the lifetime of the Product. The inputs for the design come from different

analysis sources such as reliability, maintainability, testability, or PMA. The performance of analysis activities for LSA candidates can cause concerns about its cost if there are no established threshold to reduce the effort to a suitable level. It is necessary to perform a series of assessments to balance between analysis-related efforts and acquired knowledge, which is mandatory to identify adequate supportability requirements (that satisfy the user's needs and support cost constraints). The contractor must identify the related allocation criteria and rules and propose them to the customer as general guidelines tailored to meet the specific requirements of a given project.

Note

This largely influences costs for the LSA program effort and future support activities during the overall life cycle of the IUA. It is recommended that this task be performed on a preliminary basis as an initial assessment early in a project, and to harmonize the derived strategy with the customer before signing a contract.

7.1 Principles for analysis activity selection

LSA candidates must only include analysis activities considered to be an improvement to the investigation results, in such a way that the expected benefit is greater than the effort spent for the analysis activity.

Both the customer and the contractor must take into account the knowledge acquired by "lessons learned". This includes both positive and negative experiences. Based on that knowledge, LSA relevant decisions can be predetermined without further analysis activities.

It is necessary to take into account new available analysis methods where feasible. For example, a simulation result is much more comprehensive than a traditional analysis.

Note

Based on the experience of industry, powerful simulation software packages are available and expected to be of interest for LSA purposes.

7.2 Potential analysis activities

The following activities are a list of potential analysis that, depending on the kind of project, can be performed and documented in different ways.

- analysis for the identification of general LSA needs
- comparative analysis
- human factor analysis
- product breakdown and configuration assessment
- reliability analysis assessment
- maintainability analysis assessment
- testability analysis assessment
- corrective maintenance analysis, based on existing FMEA/FMECA
- damage analysis
- special event analysis
- PMA
- LORA
- Maintenance Task Analysis (MTA)
- Software Support Analysis (SSA)
- operations analysis
- simulation operational scenarios
- Training Needs Analysis (TNA)

In addition to the analysis activities listed above, it is possible to consider some other aspects, such as:

- LCC, refer to Chap 14
- obsolescence, refer to Chap 15
- disposal (consideration of environmental regulations), refer to Chap 16

For each analysis activity, the contractor and the customer must agree whether, and how, to document, deliver, comment, and assess the results. It is recommended that most of the results of the analysis activities be documented within the LSA data, but it is also possible to document them as a special analysis report, or as an official document. Prior to any analysis, it is necessary to clarify the required results and benefits from the collected data, and how the data will be used for evaluation purposes.

7.2.1 Analysis for identification of general LSA needs

This is the starting point for any LSA activity, and is the basic precondition for any of the following activities. During this phase, it is necessary to provide a justification for the analysis performance. It is advisable to avoid using valuable resources for analysis work without a clearly defined objective. The identification of general LSA needs mainly relates to contractual, customer and user interests. In order to ensure that the customer addresses properly and agrees to the general LSA needs for the project, the following aspects must be addressed:

- Consider the results determining Product use and general support data (ORD and CRD), intended support strategy and principles, and alternative solutions. Clarify which scenarios must be analyzed, how and to what depth
- consider which contractual agreements must be verified by LSA results, how and to what depth
- consider the definition of required reports concerning areas of special interest (eg, management reports, status overviews, performance measuring reports, verification reports)
- Consider trade-off analysis results to guarantee the best return of investment. This means, for example, it is possible to use other methods to discuss customer demands and achieve the goal. The alternative solution can require significantly less effort, but it must lead to a comparative result that meets the customer's requirements at a similar and acceptable level

As a result of these first considerations, the contractor and the customer must agree on the purpose, goal, depth of analysis, and documentation method for each analysis type, in order to have a common view on the performance of LSA. The LSA PP must include the result of this analysis.

7.2.2 Comparative analysis

Performance of this analysis occurs only by special request. As a precondition, comparative information at equipment, system, or Product level must be available and effective for decision making (limited to the initial LSA activities). If it is possible to define a comparative system, appropriate analyses must be performed to derive applicable comparative data. For the documentation of the results, it is necessary to identify special summary reports concerning comparative factors and data and/or support alternatives addressed to comparable LSA candidates.

7.2.3 Human factor analysis

The results of human factor analysis influence the LSA candidates for which an MTA is performed. Human factors can be a reason for special limitations (eg, concerning the applicability of a support task requiring special abilities). It is necessary to consider ergonomic aspects, as well as the definition of rules for an appropriate man-machine interface. Refer to Chap 6. Below are some aspects that can have limiting consequences for the performance of maintenance and operational support:

- ability to lift and carry heavy loads
- ability to move for a long time in special conditions

- handling of dangerous materials
- limitations of personnel employment by law and/or security restrictions

7.2.4 Product breakdown and configuration assessment

During the LSA process, it is essential to perform an analysis to develop a structured view of the Product.

- How is it structured?
- Which functional systems and subsystems are included?
- Which items (hardware/software) are installed, how often and where?
- Are there different variants of the Product to be considered, and which different configurations are applicable for which specific variant?

For more information how to develop an appropriate Product breakdown, refer to Chap 4.

In the case of significant deviating design configurations of any LSA candidate, configuration assessments are mandatory. It is necessary to examine potential changes to validity, engineering, and deviations or waivers. Configuration assessments determine the need to deviate support concepts and/or support requirements (eg, personnel, material) in order to re-assess the need for analysis activities on LSA candidates.

The customer and the contractor must clarify how to document the configuration deviations of LSA candidates.

7.2.5 Assessment of reliability analysis

Reliability analysis is a part of the support engineering activities, referred to as Reliability, Availability, Maintainability, and Testability (RAMT). It is necessary to assess reliability predictions (eg, MTBF) related to the intended use scenario of the LSA candidates. Determining reliability values is related to reliability analysis activities, but the documentation of certain results of this analysis is important for LSA purposes. Normally, a reliability analysis is mandatory for each hardware LSA candidate. Additionally, it is recommended to extend reliability predictions to all essential items of a Product breakdown and to ensure consistency of these values over all breakdown cascades.

Note

Reliability values are directly linked to the results of FMEA activities. Any concrete value, for example the failure rate or MTBF of an equipment, is particularly interesting for LSA activities, for example to calculate the frequency of a rectifying task.

7.2.6 Assessment of maintainability analysis

Maintainability analysis is a part of the support engineering activities, often referred to as RAMT. For each LSA candidate, it is strongly recommended to perform a maintainability analysis to determine, from a technical point of view, whether and how it is possible to support an item. It is necessary to establish rules for related conditions and time frames and investigate alternative repair solutions, if applicable. Where applicable, maintainability analysis addressing the following aspects must be performed:

- assessment of maintainability features reflecting customer requirements
- identification of maintenance and operational support tasks to develop an appropriate support concept for each applicable LSA candidate
- investigation of supportability aspects such as modular design of the Product, good accessibility, installation concept in special zones (the item with the worst MTBF must not be installed behind other items)

In general, it is possible to perform the maintainability analysis by following applications with different level of detail and effort:

- best engineering judgment in the case of lowest level of available information

- assessment with or without the need to transform information on the equipment provided by the supplier. In case information from suppliers is required, apply data sheet templates to ensure that every supplier will be able to deliver the required information correctly and completely. Harmonize these templates with the customer and the contractor (and contractor suppliers).
- take into account supporting analyses such as LORA with a view to compare alternative support solutions on different maintenance levels or PMA methods, such as S4000P, in order to complete the support concept for the IUA
- use simulation tools for Products performing missions according to operational profiles, focusing on maintenance and support while evaluating potential consequences, for example from limited maintenance resources or changed operational profiles

Depending on the different types of items, it is necessary to perform the analyses given in Table 11 as a minimum level of analysis:

Table 11 Depth of maintainability analysis depending of item type

Item	Maintainability analysis depth
Items currently used under comparable conditions	Moderate information and data transformation, a more detailed maintainability analysis is voluntary
Items currently in use, but not under comparable conditions	Data transformation is mandatory, a more detailed maintainability analysis under new conditions is recommended
COTS items without major modification	The customer must document and agree to any non-compliance of supportability features and/or requirements
Items currently available requiring minor modification	Moderate information and data transformation, a more detailed maintainability analysis is voluntary
Items currently available requiring major modification	Data transformation is mandatory, a more detailed maintainability analysis under new conditions is strongly recommended
Newly developed items based on well-known technology	Detailed maintainability analysis is strongly recommended
Newly developed items based on new technology	Detailed maintainability analysis is mandatory

The results of the maintainability analysis require careful documentation. Therefore, the customer and the contractor must define and harmonize maintainability reports (eg, a RAMT case report). In addition to these reports, the support concept reflected by the identified maintenance and operational support tasks will include the results of the maintainability analysis.

7.2.7 Assessment of testability analysis

Testability analysis is a part of the support engineering activities, often referred to as RAMT. The main aspects for a testability analysis to consider are:

- identification of maintainability strategy aspects to be observed
- identification of overall test architecture and test principles
- identification and verification of contractual testability requirements
- evaluation of FMEA/FMECA results concerning testability information
- description of failure detection and localization methods at equipment, subsystem, system and Product level

- description of functional test requirements at all involved levels
- identification of related resource requirements (eg, personnel, test equipment, test software)

The testability analysis is closely related to the maintainability analysis. Results from testability directly influence the maintainability analysis. Any corrective maintenance depends on the ability to detect and localize the failure to be corrected. In addition, conditions must allow the performance of a functional test of an item or a complete system after the repair. The depth of failure detectability, localization, and verification is one of the most important factors influencing an applicable maintenance concept.

Since it is nearly impossible to implement testability features after the finalization of the item, it is necessary to describe and document the testability requirements for each applicable item very early during the design and development phase. Testability capability must be planned down to the level on which maintenance is intended.

For all items containing full Built-In Test (BIT) capability, a testability analysis is mandatory. It is necessary to identify and document the related testability features. A testability analysis report and a summary of the analysis are required. For items with reduced BIT capability (eg, COTS equipment) that are somehow monitored within the overall test architecture, a testability analysis is recommended. It is necessary to identify and document the related testability features.

The types of data concerning testability features relevant for Product support activities must be documented and harmonized with the customer. It is necessary to establish and verify the rules for suppliers and/or manufacturers on how to implement testability features (eg, by testability demonstrations/validations together with the customer and/or the contractor).

7.2.8 Corrective maintenance analysis

In the context of a supportability analysis process, the aim of a corrective maintenance analysis is to assess existing FMEA/FMECA results to identify the potential failure behavior of a Product and the need for corrective maintenance tasks.

A clear distinction is made between functional and technical failures. A system FMEA identifies/analyzes functional failures, while a so-called equipment or technical FMEA approach identifies/analyzes technical failures.

Note

SAE ARP5580 provides a detailed description of system FMECA and equipment FMECA methodology. S4000P describes the system FMEA applied for PMA purposes.

7.2.8.1 System failure modes and effects analysis

A system FMEA analyzes the different systems and subsystems of a Product using a top-down approach to the potential functional failures, their functional failure effects with corresponding criticality and finally, the corresponding failure causes. Refer to S4000P.

It is required to avoid potential failure effects that prove to be critical to safety, legal or environmental integrity. The same applies to other failure effects that are undesirable for operational, mission, or economic reasons.

Analysis results are the baseline for the identification of the Preventive Maintenance Task Requirements Interval (PMTRI) or, in the worst-case scenario, even for mandatory redesign requirements, if no preventive maintenance task is applicable and/or effective. Refer to S4000P.

Additionally, safety analysis activities like a Fault Tree Analysis (FTA) are based on system and technical FMEA. That way, it is also possible to identify additional safety critical failure combinations. Depending on probability values, redesign and/or preventive maintenance requirements must be defined and harmonized with the PMA results. Refer to S4000P.

7.2.8.2 Technical failure modes and effects analysis

A technical FMEA uses a bottom-up analysis approach to identify in detail the consequences and the probability of technical failures, which can occur to any equipment within a Product. The technical FMEA results help determining corrective maintenance tasks, such as repair or complete replacement of equipment, including task frequencies. Chap 7 describes how the technical FMEA results are used for LSA purposes.

7.2.9 Damage analysis

It is advisable to consider items susceptible to damage at least as partial LSA candidates. In general, damage is a special event, and it is not possible to predict the criticality or extent for every case. Nevertheless, it is useful to identify classes of damages that can be rectified and/or detected in the same way. It is strongly recommended to use analysis types that at least address initial diagnostic tasks and simple repair procedures (eg, simple repair of damages such as scratches). The result of these analyses must be documented as specific maintenance tasks. For example, the standard test and repair procedures for structural items or for electrical wiring are collected within a non-hardware chapter of the Product breakdown.

Normally, reliability values concerning damages are not available, therefore general quantitative statements are generally not possible. It can be possible to predict the required effort resulting from damages only using statistical evaluations. It is recommended to use such predictions with caution. Chap 8 describes how the damage analysis results are used for LSA purposes.

7.2.10 Special event analysis

In addition to damages, which are more or less a specific type of special event, other events can have an impact on the support concept for an IUA or for the Product. It is recommended to identify and document probable special events that require special maintenance tasks to guarantee the proper functionality of the Product for further use. Refer to Chap 8. Those special maintenance tasks are a part of the preventive maintenance that a proper task selection process must determine for each relevant special event. The selection process determines the Preventive Maintenance Task Requirements Event (PMTRE). Refer to S4000P.

Examples of special events that require maintenance tasks are, but not limited to:

- exceeding temperature limitations
- exceeding mechanical load limits (eg, over torque)
- exceeding maximum allowed speed
- operation in salt-laden atmosphere
- operation in sand-laden atmosphere
- lightning strike
- hard landing of an aircraft
- collision with external objects (eg, bird strike)

7.2.11 Preventive maintenance analysis

PMA aims to avoid critical failures. This includes critical failures concerning safety, law, environmental regulations, operation and/or mission completion and economic aspects. This analysis is mandatory for aspects relevant to safety, or concerning law and environmental regulations. It is strongly recommended in case significant economic disadvantages can be expected or if there is a risk in operating the Product or completing a mission. The PMA is an extensive analysis and the results have direct impact on the support concept and in worst case on Product design. It is necessary to document the requirements for preventive maintenance tasks identified by a PMA. It is recommended that rules be established to harmonize PMA results with the documentation method within the LSA program.

Each item for which at least one PMTRI is identified via a PMA process must become a full LSA candidate.

7.2.12 Level of repair analysis

Depending on the general support strategy, it can be necessary to decide the maintenance and/or repair level of each item. To support this decision, it is possible to perform a LORA by applying one of the available methods, which will depend on the type of LSA candidate to be analyzed, the required effort and/or the available information. Depending on LSA GC agreements, there can be different LORA approaches. Refer to Table 12.

Table 12 LORA approaches

LORA classification	Description
Simplified LORA	The best engineering judgment derives the simplified LORA by taking into account the best information available. The results of this analysis must be documented and harmonized with the customer for example by means of a "simplified LORA report".
Economic LORA (ELORA)	An ELORA approach is a front-end analysis using LORA software packages based on mathematical models of different complexity and accuracy. All required data concerning the item itself and its context of use (eg, operational scenario, spare part provision, and cost elements relevant for the support) must be available to perform this type of extended analysis.
Analysis via simulation	A simulation that considers detailed information about the IUA, its deployment and operational scenario can produce the best LORA results. To use a simulation software package, a complete set of information/data at the required depth and format is necessary. The preparation of this data can require a considerable effort.

Note

As shown in Fig 7, MTA information can be relevant for LORA activities. In general, a specific LORA data set is required to perform the analysis. This data set contains basic MTA information, as well as information about costs for IPS elements and information about the support organization on the operator's side.

After determining the support concept, the full MTA will be performed as required. It is advisable to avoid the creation of extended MTA data for tasks later performed at industry level.

7.2.13 Maintenance task analysis

The MTA is one of the central analysis activities within the LSA process. Here, the identified support tasks (preventive and corrective maintenance, as well as operational support) are described in detail using all required available information. The following list gives a short overview. For more information, refer to Chap 12.

- documentation of general information on tasks, such as preconditions for task performance, training requirements or criticality information
- assignment of maintenance tasks to the identified events (eg, failures, damages, special events, intervals or thresholds for preventive maintenance tasks)
- concise task description, including the sequence of subtasks
- identification of required task resources (eg, personnel, support equipment, spares, facilities and infrastructure, software)
- time estimations
- calculation of task frequencies

- consideration of required pre- and post-work (eg, gaining access, final test, release by inspector)

7.2.14 Software support analysis

Each item containing user-loadable data/software must be analyzed with respect to loading, unloading, transportation and documentation of existing software releases. This covers maintenance and servicing tasks applicable for:

- maintenance of hardware containing user loadable data/ software
- data and/or operational software required for preparation for use
- software support to update items with new user loadable software releases

If a Software Support Analysis (SSA) program manages software changes, releases, and updates in case of software failure (bug fixing), it is necessary to establish a process aligned with the LSA process for hardware. It is important to agree that any change in the software source code is an item modification, and that is not possible to regard it as a maintenance task in the context of a repair activity. Refer to Chap 13.

7.2.15 Operations analysis

It is necessary to analyze operational support requirements to identify and document important support tasks for the general handling of the entire Product, or the handling of equipment that is part of the Product in the context of PHST and/or servicing. It is also necessary to determine the method to perform the task and the required resources. Refer to Chap 9.

7.2.16 Simulation of operational scenarios

Performance of this analysis occurs only by special request. Simulation represents a very powerful tool to evaluate operational scenarios in combination with support scenarios. Simulation software packages offer the opportunity to analyze and optimize the planned use of a Product or of a combination of several Products (sometimes also described as a system of systems) under particular operational and support conditions. Only the combination of planned use scenario and planned support scenario allows generating meaningful analysis results. It is possible to achieve better results with a detailed description of the scenario.

Simulation analysis can require a multitude of input data at a certain level of depth. If simulation is selected as a required analysis activity, it is recommended that the examination of the required data of the simulation software package be started early in the project. The decision to perform simulation can influence the depth and quality requirements for any supportability information/data (eg, the detailed description of a support scenario can require additional effort within an MTA). Also, the creation of an operational scenario can require additional effort from the customer.

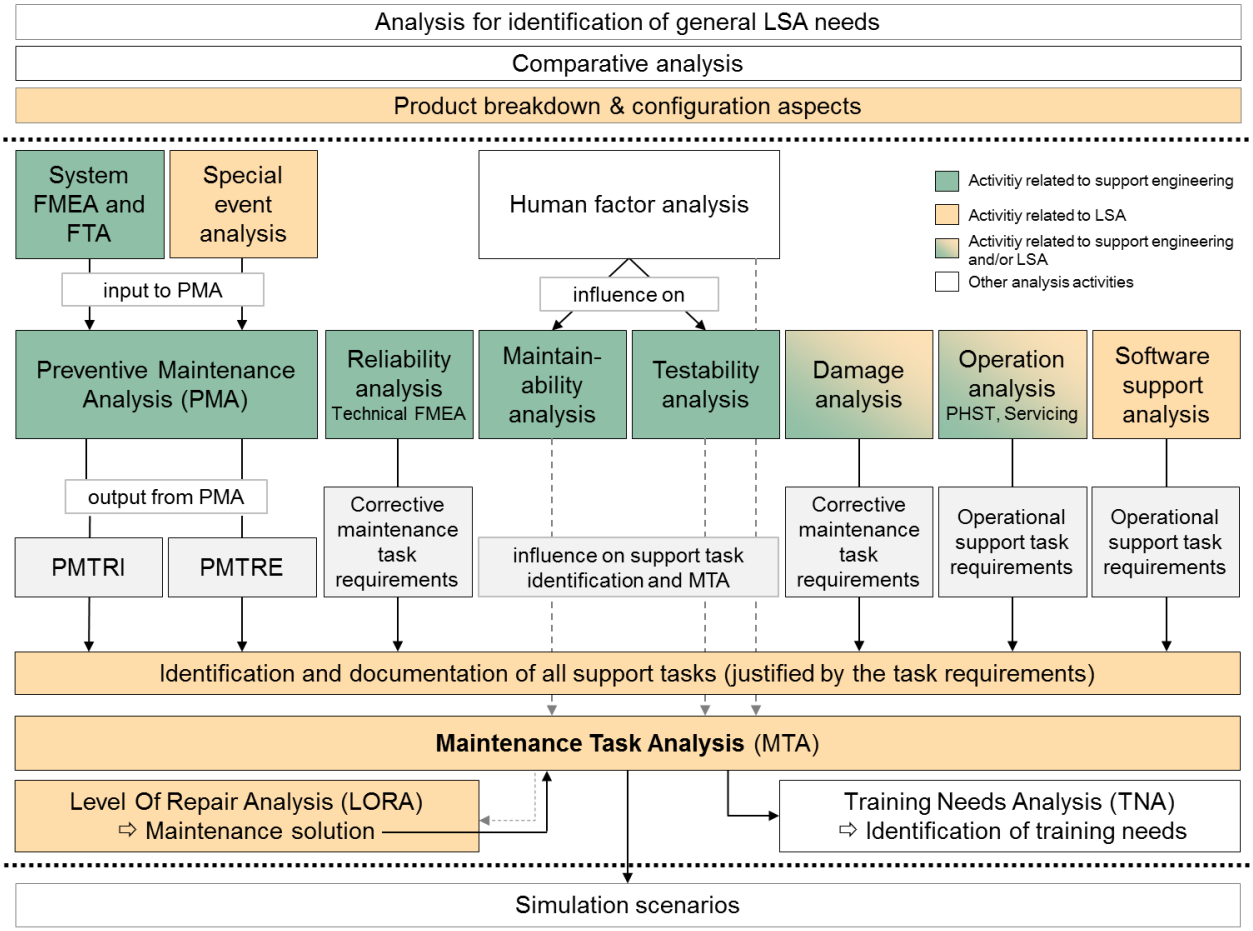
7.2.17 Training needs analysis

Within this analysis, it is necessary to determine whether a task requires special training or not. If training is required, it is necessary to determine how to provide the training in the most effective way. LSA data can effectively support this analysis process.

During the LSA GC, the customer and the contractor must discuss and harmonize the criteria for training requirements. It is possible to evaluate LSA data against those criteria to support a preliminary TNA decision. It is recommended that an IT-supported process for the TNA be established.

7.3 Analysis relations and general overview

All the analysis activities described in Para 7.2 are interconnected and can influence each other. Some analysis can be a basic precondition for other ongoing analysis processes. Some analysis activities are of general significance for all other analysis processes. Fig 7 shows a graphical representation of the relations between the different supportability analyses.



ICN-B6865-S3000L0018-002-01

Fig 7 Analysis activities relations and overview

7.4 Analysis activities selection criteria

It is necessary to consider the aspects given in Table 13 in order to select and allocate LSA-relevant analysis activities to LSA candidates that are applicable and effective depending on the type of items, candidates and required information.

During the LSA GC, the customer and the contractor must harmonize the selection criteria and aspects, and tailor them to each project by taking into account any specific circumstances.

Table 13 Analysis activities selection criteria

Criteria group	Criteria	Recommendation
Items with existing experience	Items already in use under comparable conditions	Transformation of existing analysis data
	Items already in use but not under comparable conditions	Transformation of existing analysis data
	COTS items usable without major modification	Transformation of existing analysis data
	Items currently available that would require (minor/major) modification	Transformation of existing analysis data or performance of new analysis
	Newly developed items based on well-known technology	Recommended performance of analysis activities at the required depth
	Newly developed items based on new technology	Mandatory performance of analysis activities at the required depth
Items that need to be assessed for global point of view or for special interests	Potential readiness drivers and/or cost drivers	Criteria for these items must be detailed within the LSA GC
	Of particular interest to the customer	Items must be detailed within the LSA GC
	Subject to LSA-related contractual fulfillment	Items must be detailed within the LSA GC
Items that need assessment for inherent reasons or due to the installation area	Item is significant for maintenance with respect to the related failure/defect frequency, workload, special support requirements (personnel and/or material)	Criteria for these items must be detailed within the LSA GC
	Item is subject to preventive maintenance (PMTRI or PMTRE available)	Data transfer from PMA results to LSA must be detailed within the LSA GC
	Item is potentially susceptible to damage due to its installation area or its technology	Criteria for damage analysis must be detailed within the LSA GC
Type of LSA candidates	Full candidate	Subject to applicable analysis, results documented within LSA
	Partial candidate	Rudimentary information to be documented in LSA
	Candidate families	A group of items considered as one LSA candidate, subject to applicable analysis, results documented within LSA

Criteria group	Criteria	Recommendation
	Items subject to standard procedures	Rudimentary information to be documented in LSA
Subject to request for information by the customer	Information expected from the LSA process	To be harmonized with and agreed to by the customer during the LSA GC
	Proposal of recommended support principles	To be harmonized with and agreed to by the customer during the LSA GC
	Identification of possible alternatives (for hardware, software, support concepts) and related consequences (eg, support concept, support requirements)	To be harmonized with and agreed to by the customer during the LSA GC
	Proposal of recommended support concept, including corresponding tasks	To be harmonized with and agreed to by the customer during the LSA GC
	Estimation of related support costs	Identification of related support requirements Estimation of related support costs Information for early decisions on system/project (significant costs influence)

7.5 Checklist for analysis activity recommendation

7.5.1 Analysis activities recommendation sheet

For the LSA GC, the contractor must prepare a recommendation matrix, based for example on the CIL, identifying the LSA candidates, the selection criteria for analysis activities and some preliminary recommendations, if applicable.

7.5.2 Analysis activities decision sheet

The recommendation matrix will be harmonized and finalized during the LSA GC, and reflects the decisions made by customer and contractor. The recommendation matrix can be a part of the CIL. It must be a living document in order to reflect changes, incorporate the results of the lessons learned, and enable traceability during the life cycle phases of the project. It is recommended to include in the commercial proposal a preliminary recommendation matrix including the best information available to provide the customer with a general guideline for LSA activity allocation and reduce risks.

7.5.3 Selection example

To identify the relative importance of the allocated analysis activities, it is possible to agree on the ratings concerning the importance of the selected LSA candidate, as well as the importance of the potential analysis activity. The result helps ranking LSA candidates and analysis activities from mandatory to voluntary. Such a ranking can be used to refine the selection of candidates/analysis activities, for example in case of limitations of budget or time constraints. The results of this selection procedure will compose the initial issue of the LSA CIL, as well as the range of LSA activities considered to be relevant and effective for the LSA candidates. The industry partner companies must identify and harmonize appropriate rules, and the customer must agree to them during the LSA GC.

CANDIDATE ITEM LIST (CIL) Recommendation sheet for analysis activities selection (based on ASD/AIA Mountain Bike example)				Applicable/selected supportability analysis activities 0 = not applicable 1 = recommended 2 = required											
Part identifier	Item type	Candidate Type	Element Name	Product breakdown and configuration assessment	FMA	Damage Analysis	Special Event Analysis	Level of Repair Analysis	Maintenance Task Analysis	Software/Data Loading	Software Support Analysis	Operations Analysis	Training Needs Analysis	... (further analysis activities)	Sum of rating
MTB-2000M	Product	not applicable	ASD Mountain Bike	2	0	0	2	0	2	0	0	2	1		9
└ + MTB-STS800	System	Partial candidate	Steering system	2	0	0	0	0	2	0	0	0	1		5
└ + MTB-DTS800	System	Partial candidate	Drive train system	2	0	0	0	0	2	0	0	1	1		6
└ + MTB-FRS800	System	Partial candidate	Frame system	2	0	0	0	0	2	0	0	0	1		5
└ - MTB-WHS800	System	Partial candidate	Wheels system	2	0	0	0	0	2	0	0	1	1		6
└ + MTB-WHS800-401	Subsystem	Full candidate	Front wheel	2	0	1	1	0	2	0	0	0	1		7
└ + MTB-WHS800-402	Subsystem	Full candidate	Rear wheel	2	0	1	1	0	2	0	0	0	1		7
└ - MTB-BRS800	System	Partial candidate	Brake system	2	0	0	0	0	2	0	0	1	1		6
└ - MTB-BRS800-801	Subsystem	Full candidate	Front brake	2	0	1	1	2	2	0	0	1	1		10
└ + MTB-BRS800-401	Assembly	Full candidate	Front brake lever (assembly)	2	2	1	0	2	2	0	0	0	1		10
└ + MTB-BRS800-403	Part	Full candidate	Front brake tube	2	2	1	0	0	0	0	0	0	0		5
└ - MTB-BRS800-405	Assembly	Full candidate	Front wheel brake (assembly)	2	2	1	0	2	2	0	0	0	1		10
└ + WB-10000-401	Assembly	Full candidate	Brake caliper assembly	2	2	1	0	2	2	0	0	0	1		10
└ + WB-10000-403	Assembly	Full candidate	Brake disc assembly	2	2	1	0	2	2	0	0	0	1		10
└ ISO4762 M8x60-A2	Part	Non-candidate	Caliper bolt upper	2	0	1	0	0	0	0	0	0	0		3
└ ISO4762 M8x60-A2	Part	Non-candidate	Caliper bolt lower	2	0	1	0	0	0	0	0	0	0		3
└ + MTB-BRS800-411	Assembly	Full candidate	Front brake mounting (assembly)	2	2	1	0	2	2	0	0	0	1		10
└ + MTB-BRS800-802	Subsystem	Full candidate	Rear brake	2	0	1	1	2	2	0	0	1	1		10
└ + MTB-GES800	System	Full candidate	Gear system	2	0	0	0	0	2	0	0	1	1		6
...															

ICN-B6865-S3000L0019-003-01

Fig 8 Example of a recommendation sheet for analysis activities

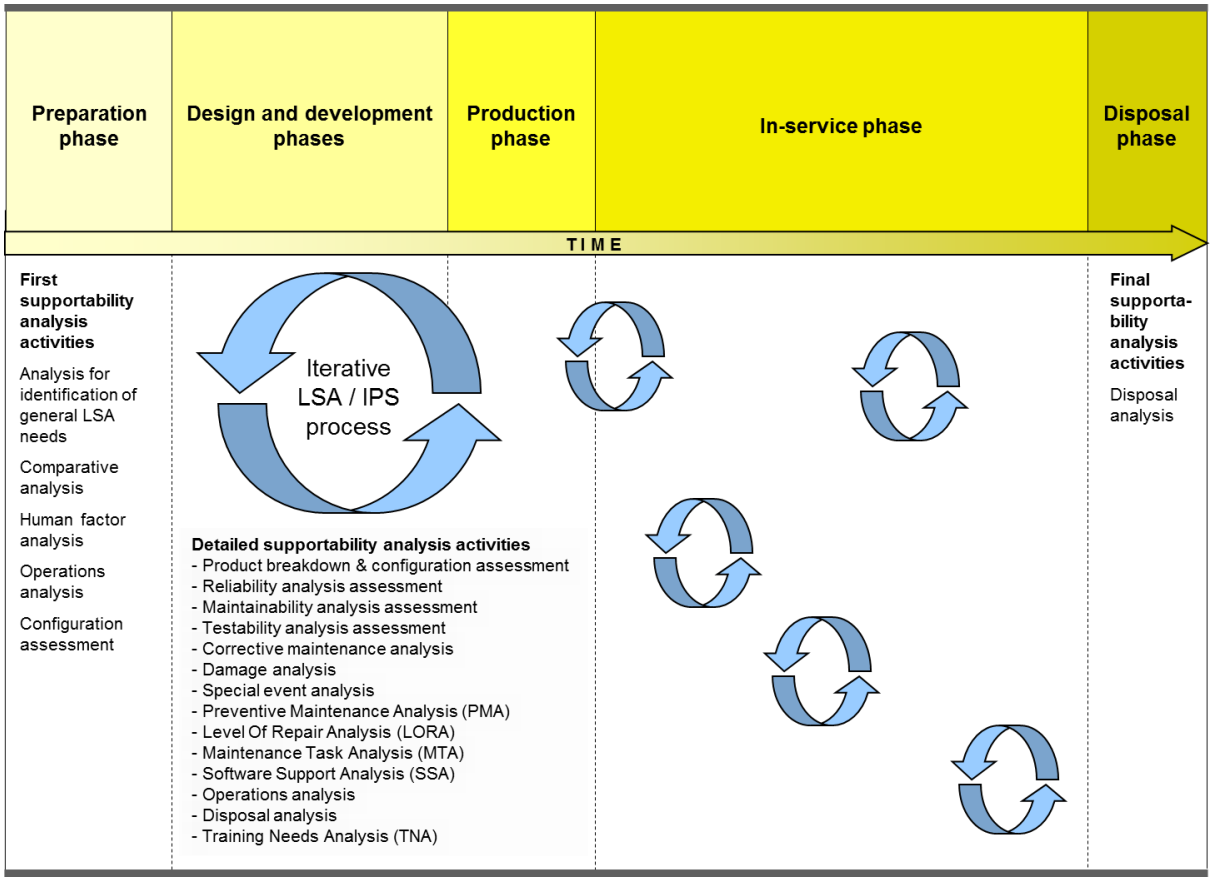
Fig 8 shows an example of a tailored recommendation matrix for the selection of analysis activities (based on a simple mountain bike example). For a real project, it can be necessary to add more analysis activities or further details. For example, a special software package can help divide the performance of a LORA into categories, such as simplified LORA or ELORA. Additionally, it is possible to provide further details on the rating values.

Note

If considered as part of the LSA program, other considerations such as obsolescence, LCC or disposal analysis can be added to the matrix.

7.6 Analysis workflow processes

In the different project phases, different sets of data/information are available for the execution of the supportability analysis activities. The performance of some of the activities described can occur during an early phase of the project. Alternatively, the performance of some of the activities can only occur when detailed and final information on the design is available. For example, it is possible to carry out an extensive MTA during a later stage of design, because this analysis requires detailed information. Refer to Fig 9.



ICN-B6865-S3000L0020-003-01

Fig 9 Position of LSA activities in the overall project schedule

Supportability analysis activities continue after the end of the design and development phase. Within the next phases (production, in-service), supportability analysis can be required multiple times, depending on the modifications being made to the Product.

This is especially relevant for the in-service phase, which is normally the longest phase. There will be extensive modifications during the in-service phase and, depending on the modifications, supportability analysis will be necessary again at the corresponding depth. As a result, the iterative LSA/IPS process will reappear repeatedly. Therefore, it is recommended that supportability data and decisions be documented throughout the entire project, ending with the disposal phase. Even for this final phase, it is necessary to consider supportability aspects.

Note

In many projects, it is not possible to draw a clear line between the phases. As a matter of fact, design and development phases and production can significantly overlap, and the same holds for production and in-service phase.

7.6.1 Supportability analysis activities in the preparation phase

In the preparation phase of a project, the supportability analysis activities can be limited to those requiring a low level of information. Before the LSA GC, information about the operational and customer requirements is usually made available, and drafts for basic rules for the LSA process are prepared.

At this stage, detailed information on the Product or the IUA is usually not available. Basic conditions and rules for the application of the process must be available during the early stages.

Nevertheless, some analysis activities can, or even must, start during the early stages of the LSA process:

- analysis for identification of general LSA needs
- comparative analysis
- human factor analysis
- operations analysis
- first concepts for Product breakdown methodology and configuration assessment

7.6.2 Supportability analysis activities in the design and development phases

During the design and development phases, the LSA process accompanies the design process. This applies to almost all LSA activities, except for the activities that must be performed in a very early phase or after a design freeze. The activities that must accompany the design and development phases are:

- Product breakdown and configuration assessment
- reliability analysis assessment
- maintainability analysis assessment
- testability analysis assessment
- corrective maintenance analysis
- damage analysis
- special event analysis
- PMA
- LORA
- MTA
- software and data uploading/downloading/transportation analysis
- SSA (software design and software maintenance)
- operations analysis

Sometimes, the collection of supportability relevant data is almost complete after the design freeze. Therefore, analysis activities that need many detailed data must be performed at a later stage of the design process, to guarantee a fixed design and reduce the risk of repeating costly analysis activities. This applies to:

- simulation of operational scenarios
- TNA

7.6.3 Supportability analysis activities in the production phase

Because the improvement of a Product by ongoing design and development activities does not stop when the production phase begins, the supportability analysis activities also continue during the production phase. Design and development phases and production phases often overlap. For that reason, although the production phase has already started, it is necessary to ensure an ongoing supportability analysis process during the production phase as well, to take into consideration any potential design change to the Product. In some cases, production needs can also cause design changes which must be considered.

7.6.4 Supportability analysis activities during the in-service phase

During the in-service phase, the supportability analysis activities will repeat to a lesser extent. If the Product has technical upgrades or design changes, it is necessary to perform new analyses concerning supportability requirements. The depth and range of these analyses will depend on the type of IUA and the degree of change.

As a second aspect, it is recommended to monitor an established support solution with respect to efficiency and cost. For this purpose, optimization processes can be performed as the In-Service Maintenance Optimization (ISMO), refer to S4000P, and the In-Service Support Optimization (ISSO), refer to Chap 17.

7.6.5 Summary of activities over the life cycle phases

Table 14 shows a summary of the potential analysis activities over the entire life cycle phases (eg, data and information collection, management, and technical/supportability analysis and preparation tasks). In this table, the design and development phases additionally contain review thresholds relevant to each phase, for example a Preliminary Design Review (PDR) and a Critical Design Review (CDR).

Table 14 Summary of LSA activities during the life cycle phases

Preparation phase	**Design and development phases**			**Production phase**	**In-service phase**	**Disposal phase**
	Feasibility study	**Preliminary Design Review (PDR)**	**Critical Design Review (CDR)**			
Performance Product data (CRD)	Performance Product data updates (CRD)					
Product use data (ORD)	Product use data updates (ORD)					
	M-studies	M-analysis	M-analysis updates		Ongoing optimization of support concept based on feedback data ⇨ in-service LSA	
	R-studies	R-analysis	R-analysis updates			
Comparative studies	Comparative studies	Comparative studies updates				
		Planning of development of IPS elements	Development of IPS elements	Update of IPS elements	Update of relevant analysis results and IPS products based on configuration changes	
					Planning of disposal	

8 Customer involvement

To ensure that the interpretations and approaches to the LSA process by the industry and the customer are in line with one another, the customer must be involved to an appropriate extent during the whole LSA program. It is necessary to cover the customer's involvement with respect to the selection of LSA candidates, relevant analysis activities and resulting analysis data, as well as the principles of information exchange, documentation of results and related management aspects.

8.1 Customer assessment of candidate items and recommended analysis activities

The selection of LSA candidates in conjunction with the determination of appropriate analysis activities must be considered as the aspect of the LSA process with the highest impact on

costs. For that reason, it is necessary to perform and assess these selections with care during an early project stage. To reduce risk, it is advisable to carry out an initial selection, or at least an estimation, concerning the expected number of LSA candidates prior to the signing of the contract.

In general, the contractor must select the LSA candidates and the related analysis activities relevant to LSA based on a Product breakdown, as detailed in Para 6 for LSA candidate selection, and in Para 7 for recommended analysis activities. The contractor must forward the results to the customer for assessment purposes and the related analysis activities be assessed between both parties to reach a well-balanced decision.

8.1.1 Establishing LSA assessment rules

The customer must agree to the assessment of the proposals, and the parties must establish some general rules concerning principles for the assessment. These rules must include at least:

- only aspects concerning rules established during the LSA GC, or rules that the customer and contractor can agree to within the subsequent LSA process (eg, during LSA reviews) must be considered as relevant for the assessment
- if an assessment has been performed and has reached a successful final clarification, there is no need to reassess it, unless new aspects have emerged and are relevant to the assessment
- it is necessary to establish rules concerning time limits between the release of LSA deliveries and an adequate response, as well as the consequences for failure to observe those rules
- Both the invitation for assessment of details within the contractor's LSA deliverables and the assessment result must be documented by using adequate status information. The contractor is responsible for the structure and information details of the status codes, but the contractor must coordinate with the customer. Refer to Para 9.4.
- any status information concerning each LSA candidate must be up to date

8.1.2 Initial assessment and re-assessments

8.1.2.1 Initial assessment

During the initial assessment, it is necessary to consider principles, such as:

- Is the Product breakdown considered in line with the rules established during the LSA GC, especially the rules concerning its structure, content and depth?
- Are rules for the selection, for non-selection or non-recommendation of LSA candidates available and sufficient?
- Is the selection of LSA candidates and relevant analysis activities in line with the rules established within the LSA GC?
- it is necessary to document the assessment results and pass them to the contractor according to the established assessment procedure, with a follow-up until a final clarification status is reached?

8.1.2.2 Re-assessments

Assessments subsequent to the initial assessment must concentrate on aspects such as:

- items released by the contractor and designated for assessment
- assessment of contractor deliveries concerning general questions or comments (new aspects)
- assessment of contractor deliveries concerning detailed questions or comments (observations)
- assessment of contractor deliveries concerning differences with respect to prior agreements

- it is necessary to document the assessment results and pass them to the contractor according to the established detailed assessment procedure, with a follow-up until a final clarification status is reached

8.1.3 Establishing an assessment procedure

It is necessary to establish a detailed assessment procedure (including an appropriate commenting process) tailored to the specific requirements of the individual LSA program and comprising the above elements. The customer and contractor must harmonize this procedure.

8.2 Information exchange between customer and contractor

To proceed with the LSA process, the customer must inform the contractor on the assessment results that are relevant to the agreements/disagreements, along with the reasons for any disagreement. In addition, there can be general comments covering global aspects and/or general questions. It is necessary to separate these comments and questions from the other comments, as they usually need a special response (eg, a dedicated explanation or request for special training).

This is not limited to LSA candidates and the related analysis activities, but it also applies to detailed results of the analysis process and to a response on any official LSA report.

8.2.1 Commenting process

It is necessary to establish a structured commenting process to facilitate the management of comments and enable traceability. This process includes as a minimum:

- registration and official comment on any LSA delivery from the contractor to the customer released for assessment
- documentation of any customer agreement
- documentation of any customer disagreement, along with the related reasons. Affected items must be monitored until final clarification has been reached and the clarification is documented.

To address these subjects, the commenting process must cover the registration of LSA deliverables, comments, and LSA status. This includes:

- Registration of LSA deliverables:
  Any LSA delivery that needs assessment must be registered according to the agreed rules

- Registration of comments:
  It is necessary to document the customer's response for each LSA delivery released for assessment. Depending on the consequences of the customer's response, each applicable LSA candidate must have a registered status.

- Registration status information:
  The structure of an LSA status code must comply with different information requirements for the released LSA candidate (refer to Para 9.4)

Note

If the customer can access the actual LSA data in the LSA program, it is recommended that the customer involvement be organized within the LSA IT environment.

8.2.2 Informing the contractor by the customer

As agreed in the established commenting process, the customer informs the contractor about:

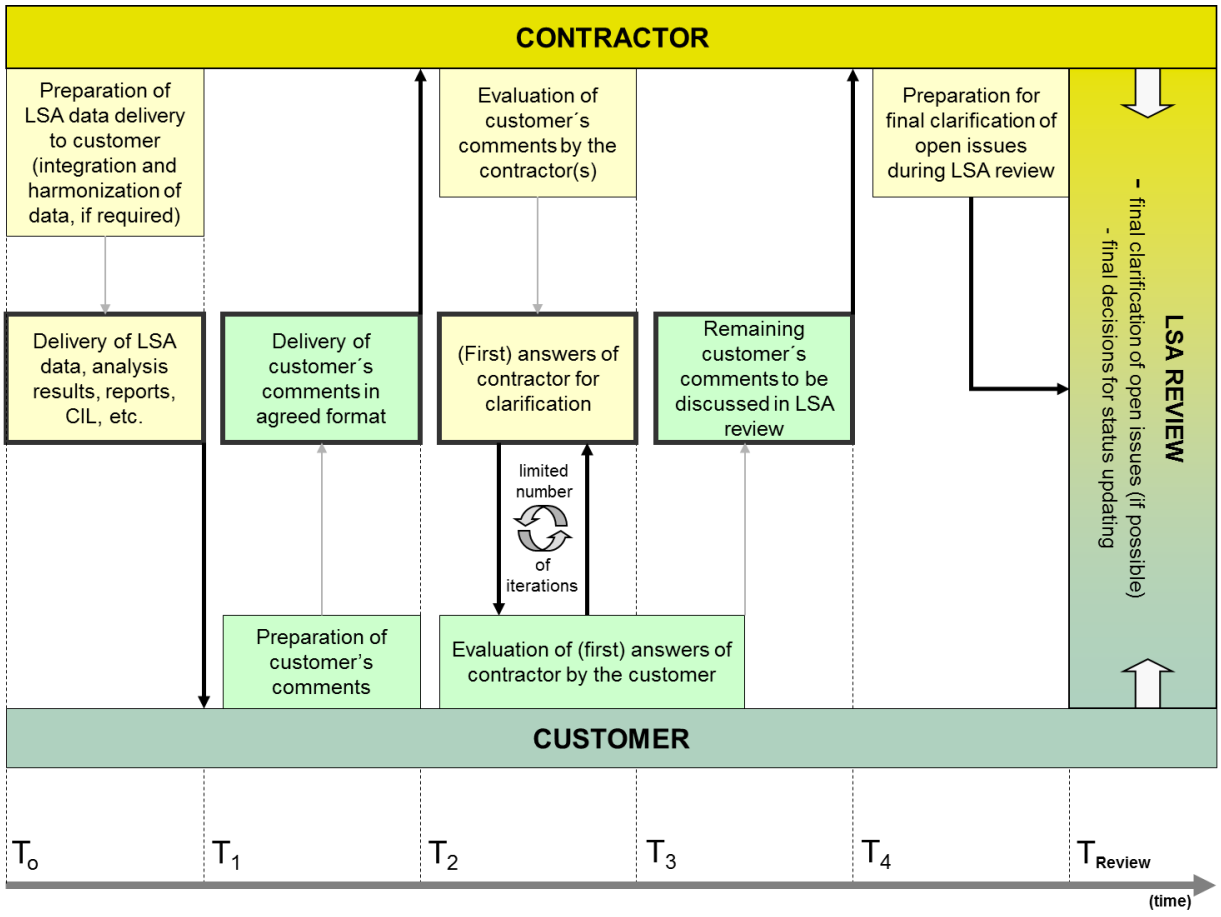
- Agreements concerning released LSA deliverables:
  The contractor will document any customer agreement according to the established management rules. In the case of a consortium, all companies involved in the contractor network must receive all the information on the related rules.

– Disagreements concerning released LSA deliverables:
In these cases, the contractor (or the involved company) will investigate the customer's disagreements to identify adequate responses (eg, further explanation, proposed rework by industry). The contractor will forward the responses to the customer in order to reach an agreement (in an iterative manner, if required).

– Unresolved issues:
In case of iterative commenting cannot resolve an issue, an adequate solution is necessary to reach a final clarification.

Fig 10 is an example of a simple process of information exchange between the customer and the contractor. Fig 10 Process of information exchange between customer and contractor (example)

Table 15 gives an explanation of the time intervals.



ICN-B6865-S3000L0021-003-01

Fig 10 Process of information exchange between customer and contractor (example)

Table 15 Explanation of schedule for the commenting process

Time	Activities
To	The contractor starts preparing the LSA delivery items. This preparation can include the integration and harmonization of data between one or more contractors, in case of a contractor consortium.

Time	Activities
T_1	Delivery of the contractual LSA delivery items to the customer in the agreed format.
T_2	Between T_2 and T_1, the customer will comment the delivered items (eg, LSA data, LSA reports). At the agreed point of time T_2, the customer will deliver the comments to the contractor.
T_3	If possible, the contractor must handle any comment that can be answered easily, before the LSA review, with a view to reducing the effort of the LSA review. Nevertheless, it is necessary to document carefully any decision concerning the customer's comments. During this time, more than one question-answer loop can clarify specific questions in detail. However, the number of loops must be limited since more extensive questions require clarification (eg, at the LSA review or by a longer clarification process between the contractor and customer).
T_4	Taking into account the clarification loops between the customer and the contractor in the time between T_2 and T_4, the customer will deliver the remaining comments to the contractor. These comments will be the basis for any discussion and clarification during the LSA review.
T_{Review}	Time of LSA review. It is necessary to ensure the documentation of every decision in the LSA review by updating the corresponding status information.

8.2.3 Final clarification on open issues

For any remaining open issues, it is necessary to identify an adequate solution. If this is not possible during a regular review session, a meeting of dedicated specialists can be required to reach a satisfactory final solution.

8.2.4 Customer decision (status influencing)

Normally, it is required to distribute an LSA delivery for assessment to different recipients (eg, different authorities in different nations and, since LSA is interdisciplinary, to all involved supportability stakeholders). Subsequently, it is necessary to collect and harmonize (in case of deviating responses) individual comments to provide one official customer response.

Note

It is important for the contractor to receive only one official customer response in order to carry out the LSA process in a manageable way. The customer must provide the decisions in the form of a status code change (following the rules established for status code allocation).

The status code allocated by the customer remains unchanged in the related customer release. Any related explanation (eg, reason for disagreement) must be documented as well, as part of the contractor response and it must remain untouched for traceability reasons.

For LSA program traceability, there are two ways to document the LSA delivered as proposed for assessment and the customer's response (if applicable, including final clarification results):

- LSA recommendation document delivered by the contractor
- LSA decision document responded to by the customer

In the case of essential documents such as LORA or PMA results, the resulting recommendation/decision documents can facilitate some clarification requirements.

Within the contractor's LSA documentation, it is necessary to update the corresponding status codes concerning the involved LSA candidate. However, contractor's information can supersede the status code allocated by the customer (eg, on-going work status, required deletions).

8.2.5 Exchange of analysis data with the customer

Exchange of analysis data must include, but is not limited to, the following steps:

- establish appropriate rules for data and document exchange - Refer to Para 8.2.5.1.
- LSA data and document collection by the contractor - Refer to Para 8.2.5.2.
- delivery of LSA data/documents by the contractor - Refer to Para 8.2.5.3.
- customer assessment and distribution of assessment results - Refer to Para 8.2.5.4.
- distribution of customer responses by the contractor - Refer to Para 8.2.5.5.
- distribution of contractor responses on rejected comments - Refer to Para 8.2.5.6.

8.2.5.1 Establish appropriate rules for data and document exchange

During the LSA GC, the customer and the contractor must document and determine appropriate agreements and rules. For example:

- software to be used
- data and report formats (eg, XML-format for LSA data and MS Excel for reports or evaluations)
- periodicity
- other reasons for delivery
- distribution partners
- binding sequences

8.2.5.2 LSA data and document collection by the contractor

During the LSA GC, the customer and the contractor must document and determine appropriate agreements. This includes, but is not limited to:

- establish an appropriate data/document exchange and file network within industry
- collection of data/documents for intended delivery between associated companies and LSA-related disciplines
- performance of industry internal quality checks, harmonization and agreement for delivery

8.2.5.3 Delivery of LSA data/documents by the contractor

During the LSA GC and the subsequent LSA agreements, the customer and the contractor must document and determine appropriate agreements. This includes, but is not limited to:

- distribution of the LSA deliverable (directly or via a designated management unit) by taking into account the agreed distribution rules including the due date, if applicable
- contractor internal documentation and distribution of the official LSA delivery

8.2.5.4 Customer assessment and distribution of assessment results

During the LSA GC and the subsequent LSA agreements, the customer and the contractor must document and determine appropriate agreements. This includes, but is not limited to:

- distribution of the contractor's LSA delivery, as required
- harmonization of the individual responses to one official customer response
- distribution of customer response to the contractor

8.2.5.5 Distribution of customer responses by the contractor

During the LSA GC and the subsequent LSA agreements, the customer and the contractor must document and determine appropriate agreements. This includes, but is not limited to:

- registration of the official customer response

- distribution within the industry
- identify, distribute and harmonize the contractor investigation results
- update of the LSA data according to the customer response details (as much as possible)
- preparation of harmonized contractor response to general comments and disagreements

8.2.5.6 Distribution of the contractor response on rejected comments
The steps concerning iterated analysis data from the contractor are applicable. Para 8.2.5.3 describes them.

9 LSA review conference

The contractor and customer personnel must take part to the planning of the LSA Review Conference (LSA RC), in order to:

- reach final clarification of open issues
- reach customer acceptance of open issues
- monitor the status of LSA candidate items (eg, completion, quality)
- assess supportability aspects related to each phase of the process
- reach additional agreements between the customer and the contractor concerning LSA aspects to improve the LSA process, as required

9.1 LSA review process

The LSA RC must be conducted at LSA candidate level. Prior to the conference, the contractor must inform the customer about the LSA candidates under discussion.

During the LSA RC, it is necessary to assess and take decisions on the related LSA deliveries and the relevant data elements, by taking into account established acceptance criteria. Any decisions must be documented.

Depending on the type of information available at different project stages, complex LSA CI can require a sequence of analysis activities distributed over a long period. On the other hand, some decisions must be made very early in the project, based on limited information (eg, establishment of a preliminary support concept), while other decisions require a stable design that is not typically available until a very late stage (eg, final task description in the context of an MTA). Therefore, it is possible to establish a set of review steps to enable sequenced decisions in line with the structured analysis process. Furthermore, the overall review process can be divided into different review steps, each supported by coherent information. Review step indicators can help identify the review steps (refer to Para 9.3). It is necessary to define the data/information required for each review step.

Due to the iterative nature of an LSA process, any approval given during an LSA RC must be considered a temporary assessment, but it must allow the contractor to continue with the LSA process. Because of the iterative nature of LSA, it is possible to re-examine the data due to changes in the design, the updating of technical data and/or the support or operational environment.

The final results documented within the LSA data (eg, at the end of each project phase) must be considered as fixed. During the LSA GC, the parties must agree to the related acceptance criteria.

The evolving design and the progress of the intended analysis activities must be the basis of the LSA RC. These meetings must take place regularly within an agreed time frame or depending on the amount of information to be assessed. Minutes of each LSA RC must be prepared to record the results and give evidence of the actions that the customer and the contractor must take. Updating related LSA data in accordance with the results of an LSA RC is a required iterative activity.

9.2 Subject for review

In general, during an LSA RC the review will concern any contractual item agreed upon during an LSA GC, or items agreed upon in subsequent agreements entered into by the customer and the contractor. In addition, during an LSA RC it will be possible to discuss any aspect that can be identified as essential for the LSA process or that could influence other support processes with interfaces to the established LSA program. The aim is to determine modified or additional LSA activities. The customer and the contractor must coordinate any occurrence within the LSA process that require common assessment in order to correct any essential problem.

The contractor must provide the customer with introductions to procedures and/or principles of LSA relevant analysis required. This will help reach a common understanding of the goal of analysis activities, related restrictions, and essential LSA results including the introduction to global analysis work flowcharts. Refer to Fig 3 and Fig 4.

The contractor must provide progress reports and assessment summaries on data quality, as well as summary reports reflecting useful information for the customer. Said reports can include, for example, a set of supportability KPIs:

- mean man hours per operating hour
- failure localization capability, such as the percentage of successful failure localization to one component within an LRU (ambiguity group/level 1)
- Mean Elapsed Time (MET) for support tasks that occurred within specified limits, along with the related percentage (eg, 90% of LRU replacement tasks must be performed in less than 2 hours)

9.3 Example for review structuring

The structure of a typical review process can have a series of steps. This way, it is possible to cover different aspects and/or types of information at different stages of the overall analysis process. The arrangement of the steps will depend on the project requirements and will realize LSA data release and acceptance within the LSA review process. Example:

- LSA review step - CIL and maintenance analysis allocation. Refer to Para 9.3.1
- LSA review step - Identification of task requirements. Refer to Para 9.3.2
- LSA review step - Maintenance Task Analysis (MTA). Refer to Para 9.3.3
- LSA review step - LORA results and support concept information. Refer to Para 9.3.4

9.3.1 LSA review step - CIL and maintenance analysis allocation

This step is the basis for all subsequent LSA activities and includes:

- BEI allocation and BE realization by corresponding parts. Refer to Chap 4
- LSA CI selection including grouping candidates, where applicable
- identification of the type of candidate and related attributes, such as:
  - identification of LRU
  - potential cost drivers
  - potential maintenance drivers
  - potential readiness drivers
- allocation of LSA relevant analysis activities to be performed for each selected candidate
- status code providing details
- management and documentation of any update regarding changes in the design configuration at CI level

9.3.2 LSA review step - Identification of task requirements

In this step, the support engineering analysis results help identifying the task requirements. The task requirements cover at least:

- corrective maintenance task requirements, such as requirements justified by FMEA results
- PMTR, such as requirements justified by PMA results
- operational support task requirements, such as requirements justified by PHST analysis results

9.3.3 LSA review step - MTA

In this step, the support tasks based on the identified task requirements will be analyzed to determine the methodology and resources needed for task performance:

- description of the required maintenance tasks in applicable detail and depth
- duration of each task step
- identification of related resources required to support the maintenance activities (eg, personnel, support equipment, spare parts, consumables, facilities and infrastructure, data and software)

9.3.4 LSA review step - LORA results and support concept information

In this step, a support concept recommendation will be prepared by allocating tasks to a corresponding maintenance level and relevant facilities. This task typically includes:

- support concept recommendation proposed by the contractor
- identification of maintenance tasks relevant for equipment integration into the Product, such as:
  - gain access and then restore the item to the initial status
  - remove and install equipment
  - perform functional tests at equipment and system level, as required
- substantiation concerning the proposed support concept as required (eg, LORA results, applied method or procedure, main influencing aspects to be considered, other analysis applied or assessed to perform LORA)
- customer decision on the support concept which could be rejection or alternative solution

9.4 Status code

A status code reflects the review process structure, and addresses contractual issues that require documentation within a project. Typically, status codes are assigned to LSA candidates or also to support tasks to document the progress of the analysis activities and customer's acceptance of LSA data. This part of a status code serves as an indicator of the review step progress. Additionally, status codes can include further management or technical information, such as:

- The responsibility of the companies involved. For each task group, it is necessary to identify the responsible company. This also reflects the agreed work share details concerning this subject.
- the type of LSA candidate (eg, full, partial)
- technical aspects (eg, LSA candidate contains user-loadable software and/or data)

Table 16 gives examples of codes that correspond to different status situations:

Table 16 Examples for status codes

Code	Description
X	Review step content is in working condition, assessment not requested
N	Review step is not applicable for the related LSA candidate
R	Review step content is requested for assessment and decision in the next LSA RC

Code	Description
A	Customer (temporarily) agrees to the review step content
B	Customer assesses the review step content and in principle (temporarily) agrees to it. However, minor rework is expected prior to final acceptance. In terms of contractual status, "B" can be treated like "A".
O	Customer did not agree to the review step content, which remains an open issue

The contractor can also use a status code internally in order to give a stop or go status to the IPS elements, to avoid any risk when creating the IPS products like technical publication.

During the LSA GC, the customer and the contractor must agree upon the intended review steps, as well as the related status code structure and applicable status codes.

10 Starting point and management of the creation of the IPS products

10.1 Starting point recommendations

During a design and development phase, several factors influence the starting point for the creation of the IPS products. The main driver is the required availability of the IPS products, for example technical publications or an illustrated spare parts catalogue. However, the production of the IPS products needs a mature situation concerning the basic supportability information/data provided by the LSA process. There must be a balance between the risk to create IPS products that will be rejected at a later stage, and the need to make the IPS products available in time. Design changes, modifications to the use scenario and/or the basic support concept have a significant influence on the IPS elements. IPS elements to be considered include:

- technical publications
- material support
- standard and specific support equipment
- personnel and training

Note

Facilities and infrastructure must be considered an IPS element not directly dependent on LSA results. Decisions on the need for facilities and infrastructure are typically made at an early stage of the project because the qualification and realization phases are usually longer. However, during the LSA process it is possible to develop some detailed requirements for specific equipment needed within a facility (eg, a workshop).

10.1.1 Technical publications

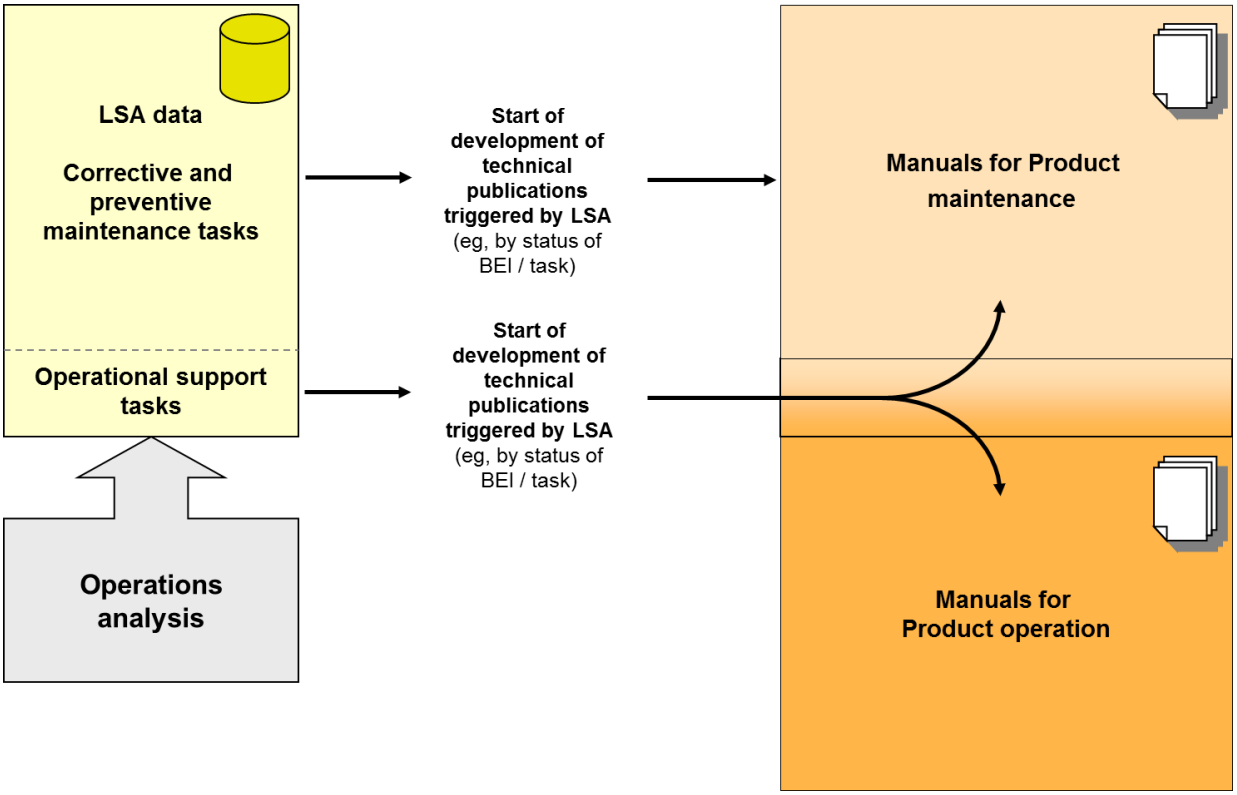
In general, technical publications consists of two main parts:

- handbooks for Product maintenance
- handbooks for Product operation, including real use (eg, how to drive a truck) and operational support (eg, how to tow a truck)

Depending on the progress of supportability analysis during a design and development process, the LSA results become more precise. It will be possible to determine the basic support concept, the required personnel, support equipment and spares, as well as details on how the task will be executed (task description). Therefore, the development of technical publications concerning Product repair and maintenance must commence at a later point. Accurate information from the LSA activities will decrease the risk of inaccurate technical publications, which can lead to rework.

In some cases, however, the development of technical publications must start early because of time constraints and contractual preconditions. The documented LSA data are the basis for the development of the technical publications providing details on Product support tasks. It is recommended that an effective solution to trigger the development of technical publications be implemented. The contractor's LSA/support engineering and the technical publications departments must harmonize this process. A good solution is to use the status information for LSA candidates, or even for support tasks, and the mechanisms for drafting technical publications as described in S1000D.

Fig 11 shows the correlation between LSA and technical publications.



ICN-B6865-S3000L0022-005-01

Fig 11 Correlation between LSA and technical publications

Generally, the development of the technical publications for the operation of the Product is independent from the LSA process. However, it is recommended that this documentation (eg, for a prototype of a technical system ready for testing and qualification) be made available as early as possible. It is particularly important to consider the information acquired from operations analysis activities. The tasks described and documented in this area can be part of the technical publications for both Product maintenance and Product operation. In this case, the recommendations for the maintenance documentation are also valid for the results of operation analysis.

The technical publications for Product maintenance and operational support must reflect each support task identified and documented within the LSA data.

However, the technical publications for Product repair and maintenance can contain further information in addition to the documented LSA tasks. In fact, sometimes it is not possible to include each single maintenance activity within the LSA data (eg, because of budgetary reasons). Not every part will be an LSA candidate. The LSA must aim to analyze, in depth, the actual drivers for maintenance and cost. It is also necessary to consider the remaining

equipment or piece parts that require technical publications. During the corresponding GC, the contractor and the customer must clarify the depth and extent of this additional technical publications.

To guarantee a meaningful information flow between the LSA and technical publications, it is recommended that the LSA and the technical publications department exchange information concerning the status of work on a regular basis. In this way, the technical publications department can be constantly updated on all relevant information needed to start the development of the technical publications. Aspects that should be covered include, but are not limited to:

- Is the relevant support task ready for technical publications?
- If the status of the LSA task is "Not ready for technical publications", what are the possible risks in producing technical publications regardless of the status?
- Is there a blocking status in LSA, which prevents the production of technical publications?
- Are there any additional activities that the technical publications must include, which are not a part of the documented LSA data (no trigger from LSA for those aspects)?

10.1.2 Material support

The identification of relevant spare parts and specific consumables is a main task within the LSA process. The identification of spare parts reflects the depth of the breakdown and the extent of the analysis activities. Depending on the support concept, the identification of spare parts and specific consumables can differ drastically. For example, in a simple 2-level maintenance scenario, the complete equipment will be replaced, and the operator of the Product will not perform any repair activity. In this case, only a small number of spare parts will be identified. However, it is possible to identify additional spare parts such as standard parts (eg, screws, nuts) as being additionally required within a replacement task of the equipment.

The depth of the Product breakdown within the LSA is a crucial point. Theoretically, a very detailed Product breakdown which even includes piece parts is possible, but the effort is huge and unacceptable from a project's point of view. Therefore, the identification of spare parts within the LSA process stops at a certain level. Typically, the maintenance and cost drivers are identified within the LSA process. This information is documented within the required resources of any maintenance and operational support task within the LSA data. For material support, it is necessary to establish a process like the process described for technical publications. Depending on the progress of the supportability analysis activities, LSA-relevant maintenance and cost drivers must trigger the creation of the IPS products concerning material support.

The same applies to technical publications, since in many cases LSA will not identify all required spare parts. However, it is necessary to include in material support all spare parts identified by an LSA task. Table 17 gives an overview of the different types of spares and their consideration in the identification process.

Table 17 List of different spare part types identified by LSA

Spare part type	Description	Correlation between LSA and material support
Complete piece of equipment	– maintenance driver – cost driver Required spare parts for (preventive or corrective) replacement tasks at operator site (replacement of complete equipment without any repair at operator site)	A maintenance task defined within the LSA must approve the identification of the equipment as a required spare part.

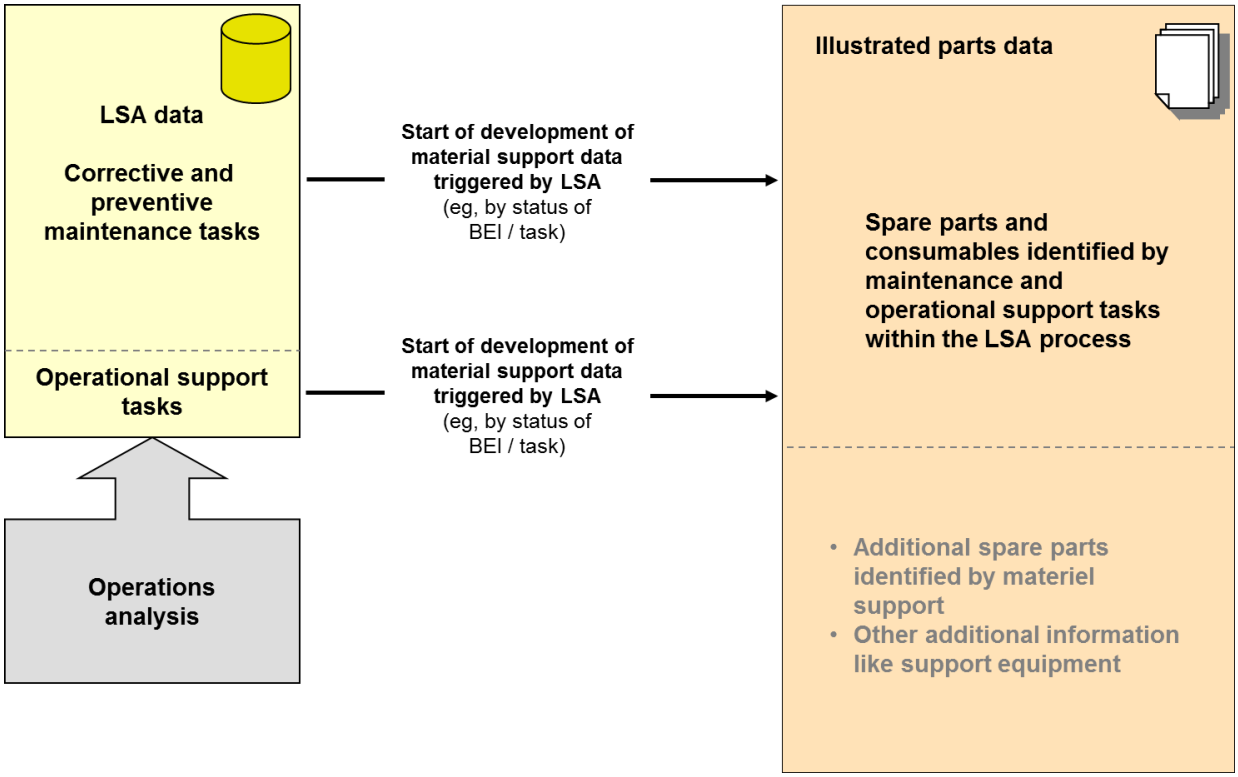
Spare part type	Description	Correlation between LSA and material support
Dedicated spare parts	Required spare parts for equipment repair or maintenance tasks at operator site (repair of the equipment will be performed at operator site)	A maintenance task defined within the LSA must approve the identification of this component as a required spare part.
Standard parts	Required piece parts (eg, attaching parts or seals) for repair or maintenance tasks at operator site (repair of the equipment will be performed at operator site)	A maintenance task defined within the LSA must approve the identification of this piece parts as a required spare part.
Consumables	Required consumables such as liquids, grease, special chemical products, source material, glue.	A maintenance task defined within the LSA must approve the identification of consumables as a required item.

Note

The dataset of IT systems for material support usually contains many more parts compared to those identified by an LSA process. The reason is that a documented support task does not identify many piece parts (eg, like screws, nuts, bolts, washers, attaching parts). Although these kinds of parts will not fail inherently, they can be lost or damaged. Therefore, it must be possible to procure those items as well.

If a maintenance task requires the replacement of any piece part, the LSA data must document said piece part as a required resource.

Fig 12 shows the process to begin the preparation of illustrated parts data within material support:



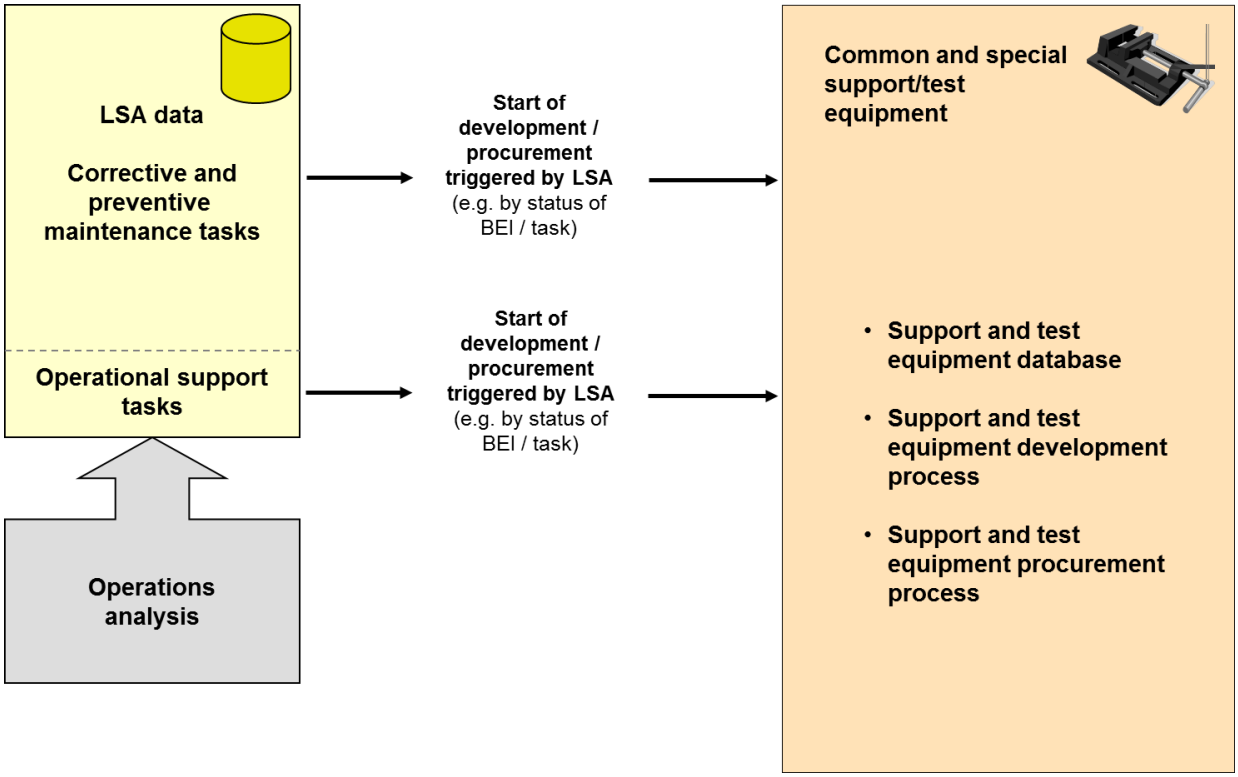
ICN-B6865-S3000L0023-004-01

Fig 12 Correlation between LSA and material support

10.1.3 Standard and specific support equipment

The identification of relevant support and test equipment is one of the main activities of the LSA process. The timely provisioning with required support/test equipment is crucial, and it is necessary to make a distinction between common and special support/test equipment. Particularly, the LSA process must identify the requirements for special support/test equipment. After the identification, a development or procuring process must begin. Refer to Fig 13.

As a requirement, each item of support and test equipment that is identified by the LSA, must be a part of a development or procurement process. The LSA status information can be used to initiate activities within the department responsible for support and test equipment. However, in some cases, sources other than the documented LSA tasks can fulfill the requirement for support/test equipment. This is valid, for example, for common tools (eg, a simple screwdriver does not need to be identified by a special LSA task), as well as for support equipment that definitely will be needed (eg, a forklift to move heavy components in case of maintenance activities). It is possible to determine the need for this kind of support equipment before the LSA data document the first support task.



ICN-B6865-S3000L0024-002-01

Fig 13 Correlation between LSA and support/test equipment

10.1.4 Training

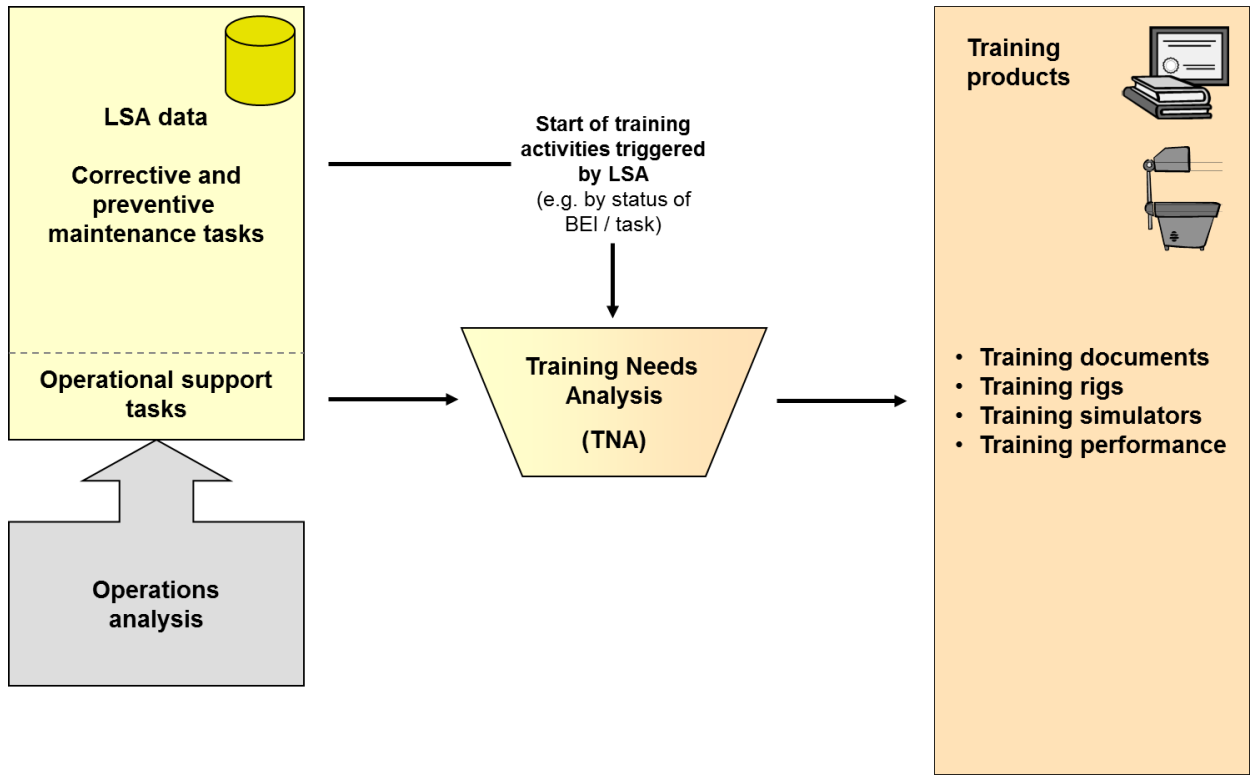
Training requires particular attention. It is possible to identify training needs concerning maintenance activities once the complete information on the required maintenance and operational support tasks is available (eg, details of performance, required support equipment, difficulty and criticality, duration, number of working steps, required personnel). TNA must be the first step in identifying training requirements. It is possible to base TNA on the support tasks identified by the LSA. An exchange of status information between the TNA and the LSA must be introduced to help identify the best starting point for activities to develop training.

Note

To support the identification of training needs not documented in LSA, and to support the development of training content, additional information should be taken from the technical publications. Refer to Fig 14.

The customer must agree to the process to identify and develop training content by taking input from LSA and/or available technical publications. The development of training equipment can be expensive (eg, production of training videos, training rigs, training simulators). Therefore, decisions concerning training must be based on reliable information.

For the best possible support, it is recommended to document the training information that is within the LSA data. The LSA data can contain simple markers, such as training required, or any other available information concerning personnel competence. Those markers ensure the extraction of training requirements from the LSA data for TNA purposes.



ICN-B6865-S3000L0025-002-01

Fig 14 Correlation between LSA and training

10.2 Management of the development of the IPS products

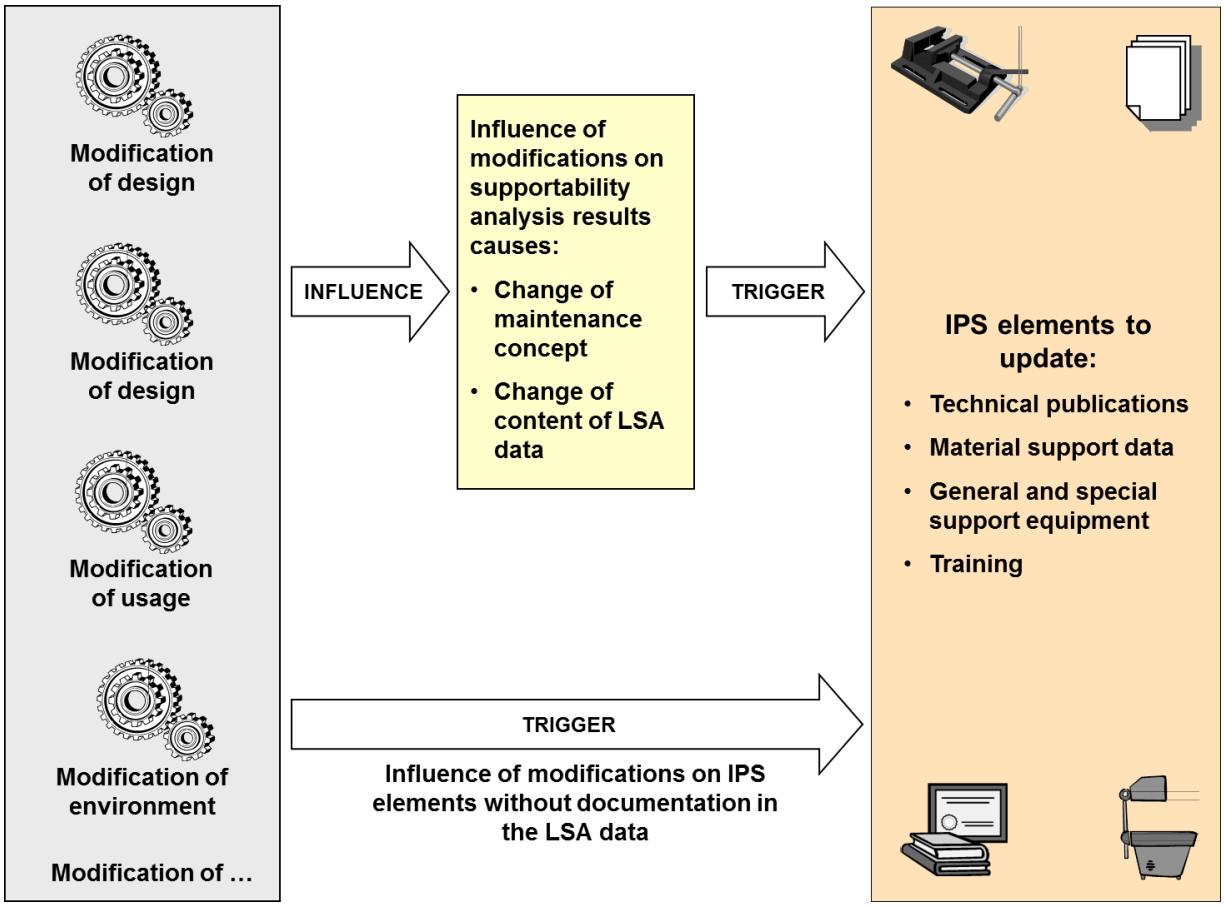
The coordination of IPS activities is one of the main challenges within the development or introduction of a new Product. Developing the IPS deliverables for new Products can be complex. Therefore, an effective solution must be at the core of the management of this development process. It is recommended that IPS managers be provided with the appropriate means to use LSA data/information as a central management tool for the entire IPS process. The contractor must consider:

- LSA status information must support the timely development of the IPS products. The support engineering and/or LSA department must generate the triggering of the IPS elements to ensure a proper start.
- any IPS element must avoid creating unnecessary effort, such as:
  - creation of technical publications for maintenance or operational support tasks that are never performed at the customer operational site
  - documentation or provisioning of spare parts or consumables that are never required at the customer operational site
  - beginning the development or procurement of support/test equipment that is never required at the customer operational site
  - planning training for maintenance tasks which are never performed at customer operational site

To avoid negative impacts of these aspects, it is recommended that a permanent quality check of the content of the IPS elements against the content of the LSA data be carried out.

10.3 Influence of modifications on IPS products

It is necessary to consider carefully design modifications that are relevant for any supportability analysis results and for the IPS elements. But not only design changes can trigger a reassessment of the existing support scenario, but also, for example, a change in the operational framework like a deployment or a new Product use scenario under different environmental conditions. It is essential to manage effectively the entire IPS process, as a common process from the design to the IPS elements. A strategy for managing modifications during every project phase is strongly recommended. Many modifications occurring as early as the design and development phases will have an effect on the entire project. Moreover, in later phases of the project, modification management for the IPS elements is vital for both the customer and the contractor. Fig 15 illustrates a typical triggering process.



ICN-B6865-S3000L0026-002-01

Fig 15 Influence of modifications on IPS elements in general

11 Checklists

11.1 Detailed checklist for the creation of an ORD

It is recommended that a checklist with a number of detailed questions be used to support the development of a complete ORD (refer to Table 18). Additionally, there can be project-specific aspects that must be taken into account.

Table 18 Checklist of detailed questions to support the ORD creation

Detailed ORD questions	Answer/Action
General use scenario	
Which are the expected uses or mission areas?	Describe use areas
Is the normal use of the Product the only possible scenario?	Yes/No?
Are there any special use scenarios, which must be taken into consideration?	Yes/No? If yes, describe special use scenarios in detail
Is permanent use expected?	Yes/No? If yes, describe in detail
Is an out-of-area use under rough conditions expected?	Yes/No? If yes, describe in detail
Is the noise from Product use or maintenance expected to have an impact on the environment?	Yes/No? If yes, describe in detail
Is the pollution from Product use or maintenance expected to have an impact on the environment?	Yes/No? If yes, describe in detail
Is the Product use or maintenance expected to have any other impact on the environment?	Yes/No? If yes, describe in detail
Are there any activities necessary to avoid the negative impact of noise and pollution on the environment?	Yes/No? If yes, describe in detail
Are there any problems concerning the performance of the Product currently in use?	Yes/No? If yes, describe problems
Are there any problems concerning the maintenance of the Product currently in use?	Yes/No? If yes, describe problems
Are there any problems concerning the supportability of the Product currently in use?	Yes/No? If yes, describe problems
Is the Product used in combination with other existing products (on a regular basis, or only occasionally)?	Yes/No? If yes, describe interaction
Does the use of the Product depend on other existing Products?	Yes/No? If yes, describe dependence
Is it possible to transport the Product?	Yes/No?
How long does it take to prepare the Product for transportation?	Duration
Does transportation require special protection and/or packaging?	Yes/No? If yes, describe in detail
Define transportability requirements (mode, type, quantity, distance, duration and conditions of transport)	Define details

Detailed ORD questions	Answer/Action
Which are the existing requirements concerning support equipment and personnel for the preparation of Product transportation?	Define details
What is the maximum acceptable time interval to recover full Product capability after transportation?	Define duration
Which are the existing requirements concerning support equipment and personnel for the recovery of full Product capability after transportation?	Define details
What other items must be transported (eg, support equipment, spares, personnel) to ensure proper Product operation at the destination and what are the requirements for these additional transportation tasks?	Define details
Deployment of locations and special conditions of every location	
How many operating locations are there?	Number
Is there an international or even an intercontinental distribution of the operational locations?	Yes/No?
What are the distances between the operating locations?	Map, distances
What are the distances between the operating locations and the required industry and/or contractor facilities?	Map, distances
Define the type of each location: – Is the location land-based? – Is the location ship-based or sea-based? – Is the location based in a mountain region? – Is the location a depot with maintenance capabilities? – Is the location a home base or an offshore location for the Product? – Is the location a training base equipped with training facilities?	Define details
Define the special conditions at each location: – Are there extreme sandy or dusty conditions? – Is the atmosphere at the location salty? – Are there extreme hot or cold weather conditions? – Are there other extreme stress conditions at special locations? – Are there limitations due to special regulations/laws at special locations? – Are there special storage requirements due to extreme weather conditions at special locations?	Define details
In a wartime scenario, are there special locations that would preclude contractor maintenance? (eg, contractors experience difficulty in performing maintenance and repairs in wartime environments)	Yes/No?
Are there any special situations involving threats that might require a special emergency support concept, with a minimum level of intervention?	Yes/No?

Detailed ORD questions	Answer/Action
Is the available infrastructure adequate to reach all locations (eg, to guarantee proper supply of spare parts or consumables)? – quality and existence of roads – airport (local or international) nearby – connection to railway stations – connection to inland or seaports	Define details
Are there special infrastructure requirements for reaching a location?	Yes/No? If yes, define details
What are the existing capabilities of each location concerning support equipment?	Define details
What are the existing capabilities of each location concerning facilities and infrastructure?	Define details
What are the existing capabilities of each location concerning personnel?	Define details
What are the planned capabilities of each location concerning support equipment?	Define details
What are the planned capabilities of each location concerning facilities and infrastructure?	Define details
What are the planned capabilities of each location concerning personnel?	Define details
Will there be plans to establish a repair station at special locations?	Yes/No?
Will there be plans to establish a supply depot at special locations?	Yes/No?
Will there be plans to exchange of support equipment or personnel between different locations?	Yes/No?
Product support and deployment	
How many supported Products are planned per location?	Define details
Deployment of Products per location	Define details
Will there be plans to share Products between different locations?	Define details
Product use overview	
What will be the normal use of the Product at every location (key use)?	Define details
What is the planned availability of the Product at each operating location?	Define details
When the Product is used in missions, what are the planned mission success rates for each operating location?	Define details

Detailed ORD questions	Answer/Action
Are there any periods of special use of the Products at a special location? – low payload period – temporary peak loads – temporary storage of the Product – out-of-area use under special or rough conditions – wartime use	Define details
In the case of permanent operational conditions, what are the possible downtimes for maintenance (maintenance windows)?	Define time periods for maintenance
Define additional Product performance parameters in measurable and quantifiable terms, such as: – range of use – required accuracy – payload values – required temperatures – required speed – required distances	Define detailed values
What is the key measurement unit for Product use per unit of time? Examples are: – operational hours for general Products – flight hours for airborne Products (aircraft, helicopters) – kilometers or miles for land-based vehicles – rounds or cycles for periodic processes in a Product – tons for transportation systems	Define detailed values
What is the expected operational profile at each location? – How many operating hours or missions are planned per day? In case there are different days for typical use, define the operating profile for each "day type". – What are the typical operational profiles for longer periods, such as weeks, months or years?	Define an operational profile as detailed as possible
How many operating days/operating hours are expected for a year or for any other basic time period?	Define detailed values
What is the average duration of each different use (eg, operation, mission, trip, flight, dive)?	Define detailed values

11.2 Detailed checklist for the creation of a CRD

It is recommended that a checklist with a number of detailed questions be used to support the development of a complete CRD (refer to Table 19). However, there can be project-specific aspects that must be taken into account.

Table 19 Checklist of detailed questions to support the creation of a CRD

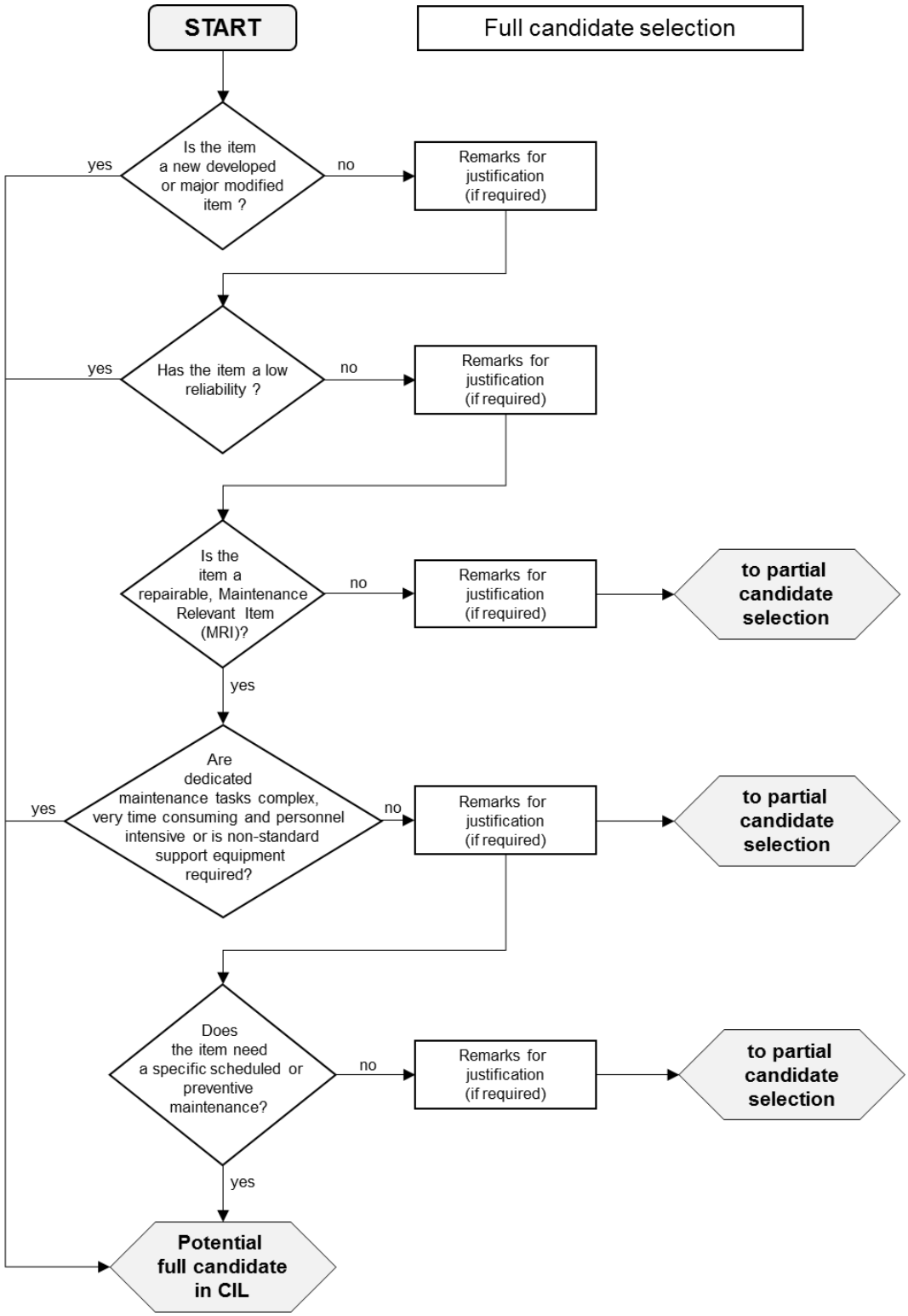
Detailed CRD questions	Answer/Action
Supply concept	
Are central depots expected, and how is their deployment planned?	Define details
Are depots planned directly at the operational locations?	Yes/No?
Is a material exchange between different locations expected?	Yes/No?
Has a selection process for suppliers been defined and how will suppliers be integrated into the expected supply concept?	Define details
Is it possible for a production line to provide spare part support (especially at early stages)?	Yes/No?
Is outsourcing via a support agency expected?	Yes/No?
What are the limitations for spare part availability and lead times?	Define details
What distribution and IT systems will be used to supply spare parts?	Define details
Which concept will be used for initial provisioning? – Is it expected to use an optimization or a simulation tool? – Is the deployment schedule properly supported by the initial spares delivery?	Define details
Will spare part delivery have an impact on the production schedule?	Yes/No?
During out-years, is the industrial base compelled to provide spares support for items that remain in the inventory?	Yes/No?
Support equipment concept	
How can you ensure that all support equipment be available in time?	Define details
What is the identification and realization process for standard and special support equipment?	Define details
Is the automated test equipment used effective to support the Product?	Define details
What are the plans for maintenance and support of the support equipment? – repair and overhaul of support equipment – calibration of support equipment	Define details
Personnel integration	
What are the general requirements and provisions for manpower and personnel?	Define details
Is it the customer or the contractor that provides staff for operating the Products?	Define details
Is it necessary to consider cooperative models?	Yes/No?
How can you ensure the timely verification of the required operator training?	Define details

Detailed CRD questions	Answer/Action
How can you ensure the timely verification of the required maintenance training?	Define details
What training processes must be developed to ensure adequate operational and maintenance support at all levels during the entire Product life? It is necessary to consider the possible rapid turnover of personnel.	Define details
Facilities and infrastructure	
Which infrastructure must be available at the required facility (eg, electricity supply, hydraulic power, compressed air, special working environment)	Define details
Are existing facilities at operational locations adequate to the requirements of a new Product (operational and maintenance facilities)?	Yes/No?
Can existing facilities at operational locations be adapted to the requirements of a new Product (operational and maintenance facilities)?	Yes/No?
Is a realistic time schedule available for the building activities for the required new facilities?	Yes/No? Define details
Is there a plan to identify and reduce risks caused by the delay in building the facilities?	Define details
IT and communication resources	
Which are the existing IT architectures at every operational location concerning the following aspects: – network capabilities – data storage capabilities – computer capabilities – communication resources	Define details
Are the existing IT and communication capabilities appropriate for the new Products that are going to be introduced?	Yes/No? Define details
Are there any risks related to future changes in IT or communication architecture that can affect the operation of the new Products?	Define details
Are there any concepts for the initial provision of data for the required IT systems? – Who holds the required data? – Are there any contractual aspects to be considered? – Are there any problems concerning data security requirements?	Define details
Are there any plans for a first installation of required new IT systems and for the appropriate service of IT systems during the life of the Product which requires support?	Yes/No?

Detailed CRD questions	Answer/Action
Are there any service contracts for hardware and software (planned and existing)? – ensure fast reaction to downtimes – the terms of any service contract must include the upgrade of hardware and software in case of technical progress	Yes/No?
Are the different IT systems at different operational locations compatible?	Yes/No?
Is there a plan to ensure the existence of all required interfaces to other IT systems?	Yes/No?
New organizational structures	
Will any organizational structure be changed at operational locations and what are the risks related to these changes?	Define details
What are the existing opportunities in structure changing for the introduction of a new Product (eg, reduction of personnel)?	Define details
Schedule consideration	
What are the plans to include IPS elements at an appropriate level in all phases of the project?	Define details
Are supportability analysts involved since the earlies phases of the project, so they can influence basic decisions concerning design?	Yes/No?
Additional aspects	
What additional aspects specific to the project are highly significant for the supportability analysts?	Define details

11.3 LSA candidate selection flowchart examples

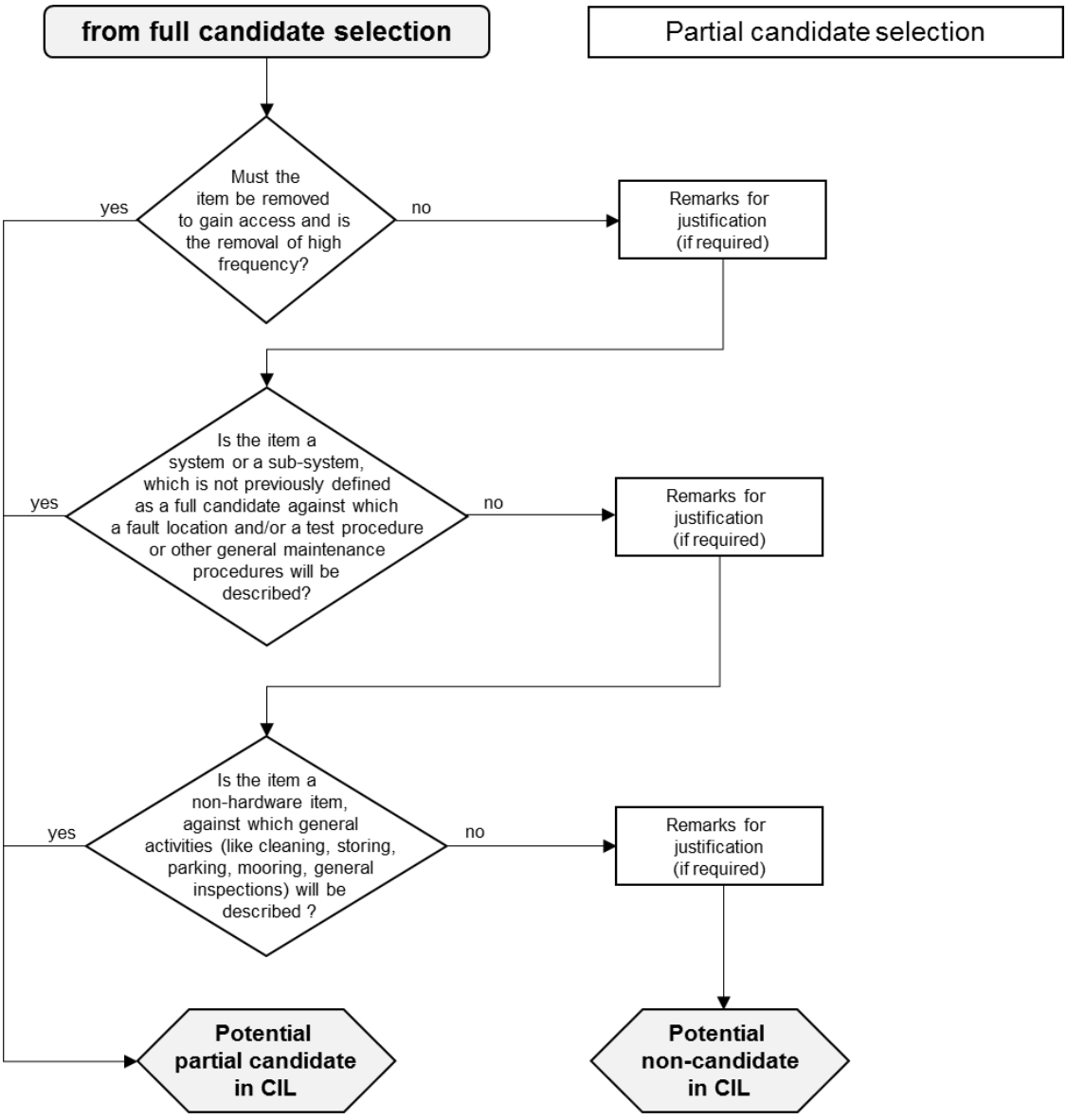
11.3.1 Non-structural items



ICN-B6865-S3000L0036-002-01

Fig 16 Full LSA candidate selection flowchart

The selection flowchart in Fig 16 shows the typical process used to determine full LSA candidates from an existing breakdown. Depending on the project, it can be necessary to adapt the selection criteria as there can be additional criteria. The flowchart must be applied to each breakdown element or part. If the selection flowchart for full LSA candidates leads to the IUA not being a full LSA candidate, it will be necessary to apply to each of those breakdown element or part the second LSA selection flowchart for partial candidates. Refer to Fig 17.



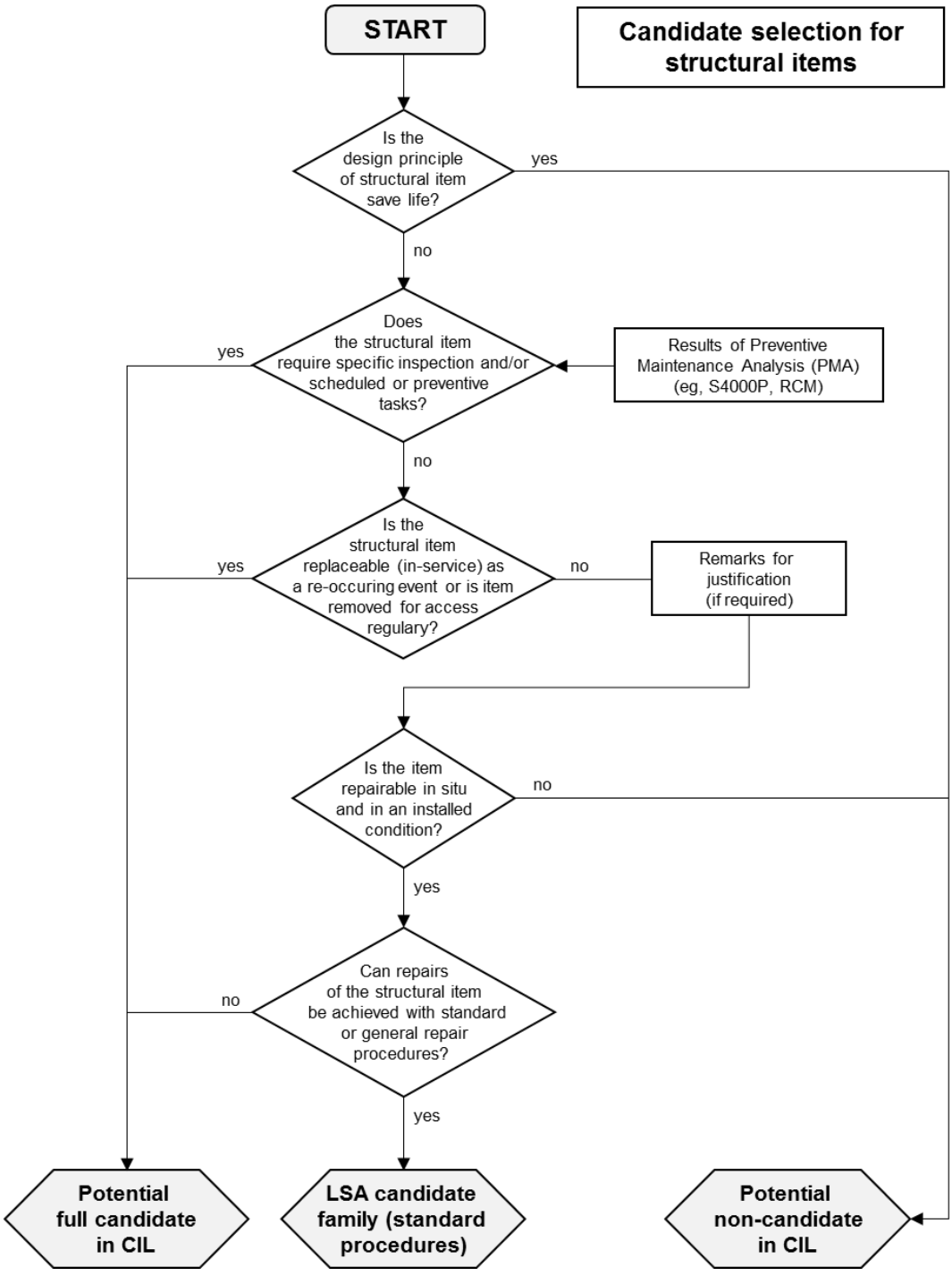
ICN-B6865-S3000L0037-002-01

Fig 17 Partial LSA candidate selection flowchart

Depending on the project, it can be necessary to adapt the selection criteria as there can be additional criteria. In general, the analyst must have a guideline for the LSA candidate selection. The contractor and the customer must harmonize any required change of flowchart logic within a project and agree to it during the LSA GC.

11.3.2 Structural items

In the case of structural components, the selection process for LSA candidates differs from the selection process for non-structural items. Refer to Fig 18.



ICN-B6865-S3000L0038-002-01

Fig 18 LSA candidate selection flowchart for structural items

12 Associated parts of the S3000L data model

The documentation of the associated data to this chapter is supported by the following Units of Functionality (UoF), refer to Chap 19:

- S3000L UoF Applicability Statement
- S3000L UoF Breakdown Structure
- S3000L UoF Logistics Support Analysis Message Content
- S3000L UoF LSA Candidate
- S3000L UoF Message
- S3000L UoF Part Definition
- S3000L UoF Performance Parameter
- S3000L UoF Product and Project
- S3000L UoF Product Usage Context
- S3000L UoF Security Classification

Chapter 4

Product structures and change management in LSA

Table of contents

List of tables

List of figures

References

Table 1 References

Chap No./Document No.	Title
Chap 1	Introduction to the specification
Chap 3	LSA process
Chap 5	Influence on design

Chap 11	Level of repair analysis
Chap 12	Task requirements and maintenance task analysis
Chap 17	In-service LSA
Chap 19	Data model
Chap 20	Data exchange
S1000D	International specification for technical publications using a common source database
S2000M	International specification for material management
S4000P	International specification for developing and continuously improving preventive maintenance
SX001G	Glossary for the S-Series IPS specifications
ISO 10303:239	Product Life Cycle Support
MIL-STD 1388-2B	DoD requirements for a Logistics Support Analysis Record
GEIA-STD-0007	SAE Handbook for Logistics Product Data (former MIL-STD 1388-2B)

1 General

1.1 Introduction

Product structures and change management ensure the correct identification of different Product configurations, controls changes, and records the change implementation of the physical and functional characteristics of the Product's structure, systems, subsystems, equipment and components.

Establishing a Product breakdown is essential to identify support Product structure for all configurations during the project life cycle. Traceability between design structure and support breakdown structure is needed in order to control possible changes in the design Product structure during all project phases and implement those modifications into the support breakdown.

Change management is another aspect that must be controlled. Along the life cycle of the Product, changes coming from different sources modify the original Product baseline structure. The LSA process must be part of the Configuration Management (CM) process to identify and analyze what was/is required, designed, produced and supported. It also evaluates changes, to identify their impact on LSA activities and implement such changes.

This chapter covers two main subjects. First, it focuses on the ways in which to define and manage Product structures with a view to organizing and recording the results for the respective LSA activity. Second, this chapter illustrates that the S3000L data model and its associated exchange format define constructs which support an iterative LSA process.

S3000L supports the definition of Product structures with varying degrees of complexity. A Product structure in S3000L can cover anything from a simple piece of equipment to an advanced Product such as a combat vehicle, an aircraft or a carrier.

It also describes how to manage support analysis activities for system of systems and Products in Products using constructs enabled by the S3000L data model and its data exchange format. Although it is possible to develop these Products at different times, and in accordance with

different product data and support analysis standards, it is also possible to incorporate them into one overall support definition.

1.2 Purpose

The purpose for this chapter is to present how the LSA process can reuse Product structures coming from design and development, and use those structures as a basis to define additional or modified Product structures that support the LSA activities. The purpose is also to illustrate how LSA is an integral part of the overall CM process and how to control and implement changes in the support solution throughout the Product life cycle. Product structures defined during LSA will also be the basis for Product structure needed by other IPS disciplines.

The chapter provides rules and guidance for the creation of Product structures for different types of Products. Below some examples are given to allow an efficient CM. The chapter also analyzes change management and the impact on LSA activities.

1.3 Scope

The target readers of this chapter are analysts who perform analysis activities within the LSA process, giving the necessary guidelines on the different available options to define the most appropriate Product structures based on the project requirements. It is also essential for traceability between design and support Product structures, and for change management control within the CM process.

Note

Figures in this chapter do not always reflect the exact representation in the S3000L data model. In many cases, they are simplified to give the reader a basic understanding of concepts and constructs covered in the specification.

2 Product

2.1 Product development and LSA

The core principles behind product structures and change management in S3000L are:

- LSA is performed in parallel with product design and development
- LSA supports a system of systems perspective

This means that S3000L is designed to meet the requirement to start LSA activities at the same time as product design, and those LSA activities will iterate as product design and development progresses.

It also means that S3000L is designed to support LSA activities to be performed against a set of Products, taking into consideration that individual Products can be in very different life cycle phases, and their respective Product can be structured and documented in a completely different way. Some Products can be newly developed, while others are mature and have been in service for some time.

Therefore, S3000L supports the idea that different principles can underlie the organization of a Product structure. However, it must be possible to incorporate Products into an overall Product and system of systems breakdown structure.

To this end, product structure definitions in S3000L share the same underlying principles as existing Product Data Management (PDM) / Product Lifecycle Management (PLM) systems and standards, such as ISO 10303-239 Product Life Cycle Support.

2.2 Use of the term Product in S3000L

A Product in S3000L (and in SX001G) is defined as a family of items, which share the same underlying design purpose.

Examples:

- Nexter VBMR Griffon

- Airbus A340
- Boeing 787
- Aegis Class Destroyer
- Ford Fusion
- Rolls Royce Pegasus Engine
- Apple iPhone 7

A Product then comes in one or many variants, and each product variant is configured for a specific purpose and made available to the market. As laid out in S3000L, each Product must have at least one defined product variant.

Examples:
- Nexter VBMR Griffon Command Vehicle vs. Medical Vehicle vs. Personnel Carrier
- Boeing 787-800 vs 787-900
- Ford Fusion S vs. SE vs. SEL

Note

In S3000L, the term "product variant" corresponds to the broader sense of the term Product. A common definition of Product is something that is offered to the market to satisfy a requirement or need.

Note

Product variant is also referred to as model in S1000D and S2000M.

2.3 Project and system of systems in S3000L

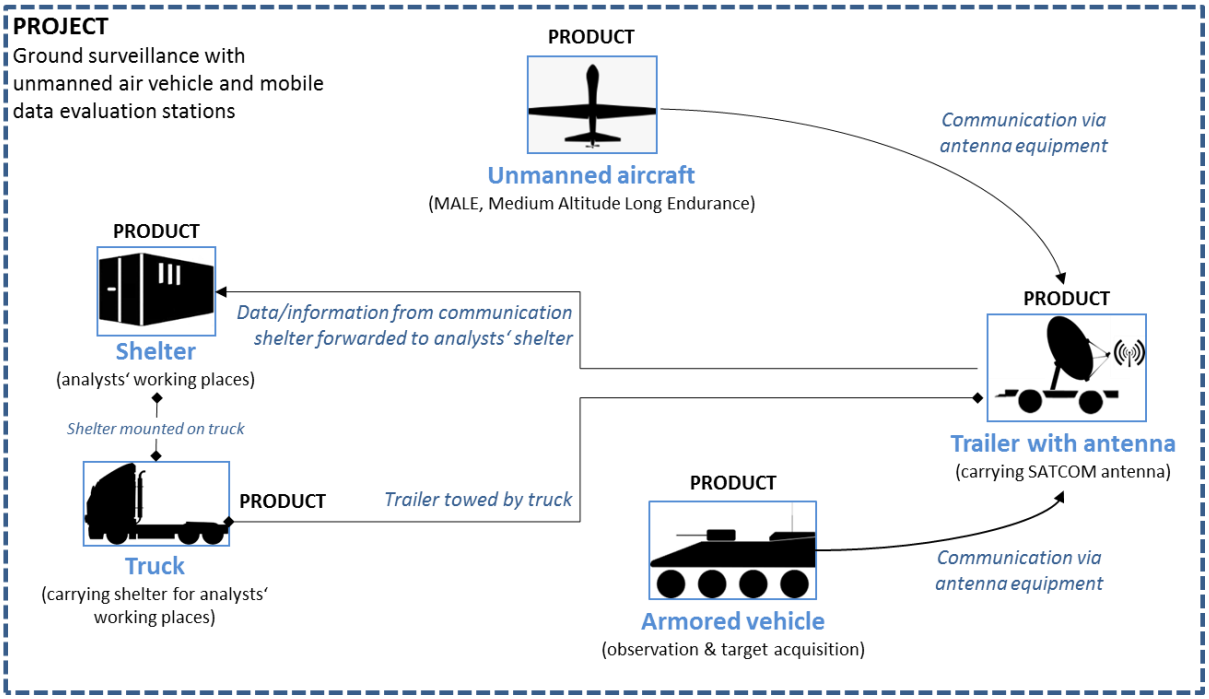
Chap 3 provides a general description of LSA projects. It is necessary to underline the importance of being able to define and manage Product structures from a system of systems perspective.

A system of systems is often defined as a collection of Products, each capable of independent operation, that interoperate together to enable more advanced capabilities. However, S3000L defines a system of systems as a collection of Products. Although each Product is potentially defined and developed independently, its respective support requirements must be harmonized against a common support solution.

The project concept in S3000L aims to organize a set of contracts and Products under one LSA project, and to meet the requirements for the overall system of systems even when performing LSA analysis activities individually for each Product.

Example

An air ground surveillance system of systems can include, for example, an unmanned aircraft, ground based antenna trailer and communication shelters (refer to Fig 1).



ICN-B6865-S3000L0115-001-01

Fig 1 System of systems example

3 Product structure

3.1 Introduction

S3000L covers two basic capabilities, which can be used to define hierarchical product structures.

The first capability allows partitioning a Product into hierarchical parent-child structures consisting of related elements. Hereafter these structures are referred to as breakdown structures, or simply breakdowns. For example, it is possible to use this capability to define a system breakdown, a functional breakdown, and/or a zonal breakdown (refer to Para 3.2).

The second capability allows managing different types of parts lists, often referred to as a Bill of Material (BOM). A part list represents the collection of parts included in the assembly of the parent part (refer to Para 3.3).

S3000L also supports the possibility to combine breakdown structures with part lists. For example, a breakdown element that represents an installation location for equipment within a Product, can refer to one or many parts that can be installed at that location. These parts can have part lists, and it is not necessary to include these part lists in the breakdown structure that includes the relevant part.

Another important aspect of S3000L is that it does not dictate any specific principles for Product structure definition for LSA. The goal of S3000L is not to impose any specific Product structure principle, since the principles for Product structure definition vary depending on application domains, companies, etc. This also includes principles and rules on how to set up the identification of elements in the defined breakdown structures. For example, S3000L does not dictate the use of Logistics Control Number (LCN) or Standard Numbering System (SNS) like GEIA-STD-0007 or S1000D, respectively. However, individual programs and projects can define project-specific business rules to use any of those, or other principles.

In addition to defining breakdown structures and parts lists, S3000L can also keep detailed information for the allowed product configurations. In the context of product variants, serialized

products and/or complex part assemblies, an allowed product configuration defines permitted combinations of breakdown elements, hardware parts, software parts, and parts list entries.

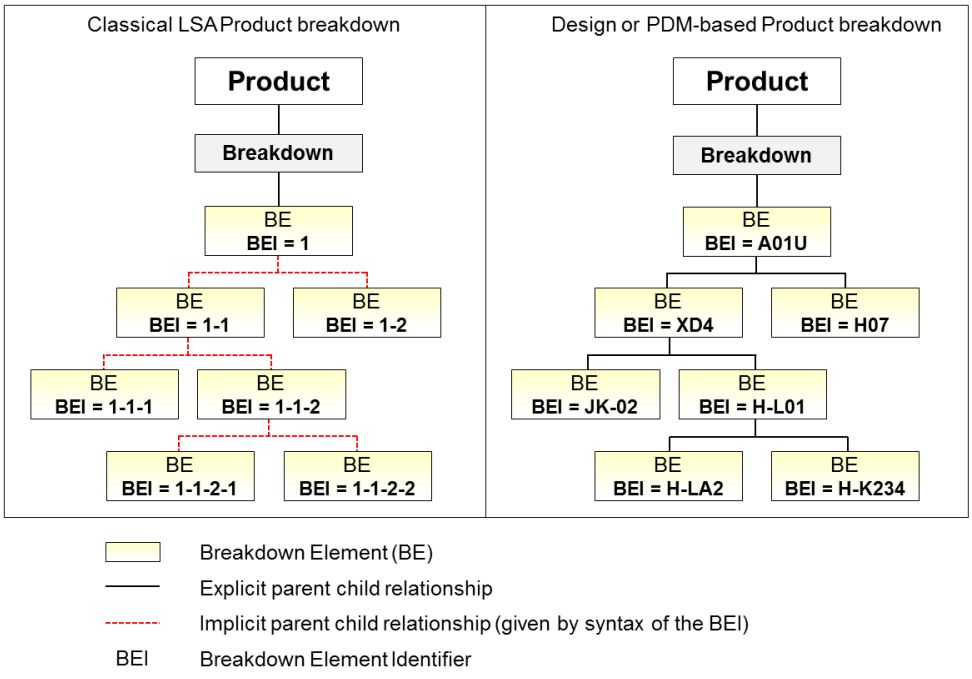
3.2 Breakdown structures

3.2.1 Breakdowns and breakdown elements

Breakdowns represent the partition of a Product or product variant into a hierarchical parent-child structure of related breakdown elements. For example, it is possible to use breakdowns to define a system breakdown, a functional breakdown, a physical breakdown and/or a zonal breakdown for a Product. A breakdown can also be hybrid, containing a mixture of system, zonal and physical breakdown elements.

S3000L has one generic capability which can be used to represent any number of breakdowns. The breakdown type attribute determines the type of breakdown being defined.

Each breakdown consists of a set of breakdown elements. Breakdown elements can be organized into hierarchical structures either by establishing explicit parent-child relationships between the respective breakdown elements, or by assigning identifiers to the respective breakdown element to indicate its position in the breakdown structure. These approaches are referred to as explicit and implicit breakdown structures, respectively (refer to Fig 2).



ICN-B6865-S3000L0082-003-01

Fig 2 Example of implicit and explicit breakdown structures

Note

The implicit breakdown approach reflects legacy IPS standards and specifications like GEIA-STD-0007 and S1000D, which use Logistics Control Number (LCN) and Standard Numbering System (SNS) respectively to indicate the breakdown structure. The explicit breakdown approach reflects current PDM/PLM approaches, which organize objects through explicit parent-child relationships.

In S3000L, a breakdown can include four different types of breakdown elements. These are:

- an aggregated element, which acts as a container for a collection of other breakdown elements. Aggregated elements are typically used to represent systems, subsystems, functions, sub-functions, families, etc.
- a hardware element, which is "realized" by hardware part(s)

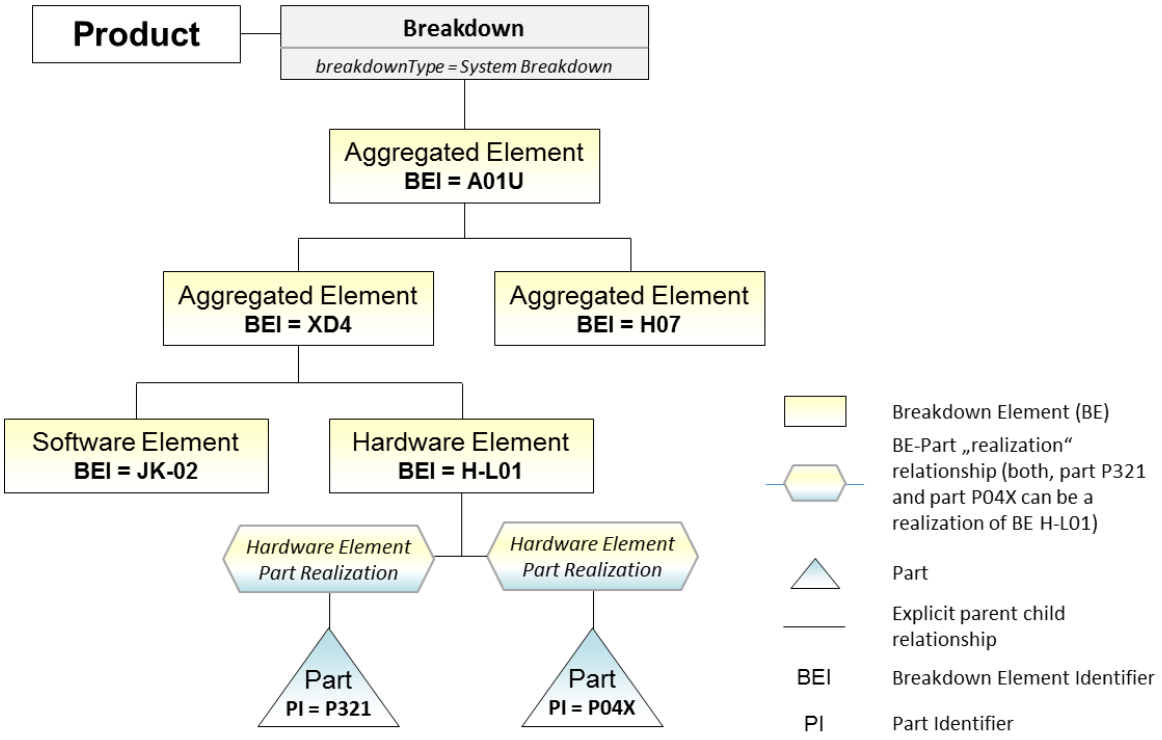
- a software element, which is "realized" by software part(s)
- a zone element, which represents a three-dimensional space related to the Product

Then, it is possible to further specialize each type of breakdown element using its respective type attribute.

It is important to note that hardware and software elements represent the use of hardware and software parts in the context of a Product. For example, hardware and software elements do not represent the hardware or software part themselves. Each hardware and software element can refer to one or many hardware or software parts used to realize the respective hardware and software element. Refer to Fig 3. If there is more than one allowed realization for a hardware or software element, these alternative realizations are regarded as substitutes (eg, as alternates in a specific usage). For more details on substitute and alternate parts, refer to Para 3.3.

Note

It is necessary to consider breakdown elements and parts as separate entities, and as having separate specifications (definition). The specification for the part must be in line with the specification for the breakdown element. If not, it is necessary to define a deviation with respect to the realization of the breakdown element (eg, the inclusion of the part as a possible realization).



ICN-B6865-S3000L0116-001-01

Fig 3 Relationship between breakdown elements and parts

It is necessary to give special attention to software since it is an integral part in most modern state-of-the-art Products. Any supportability analysis must consider software, which can require, for example, to define loadable software packages as part of the breakdown structure, if the software requires a task definition for its installation/de-installation. The Product breakdown must also define the effectivity of software and hardware combinations (refer to Para 3.5). If the hardware does not undergo any modification, but the integrated software does, it is necessary to document such modifications as being part of the Product structure.

Note

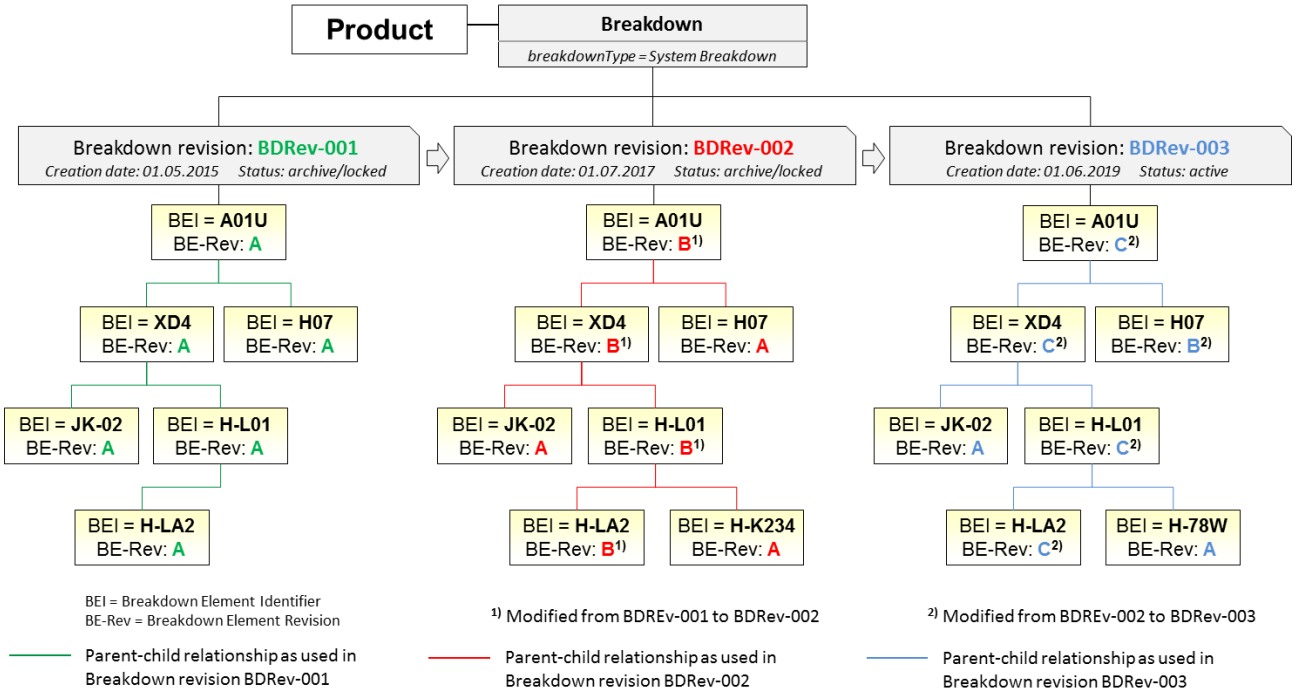
Software implementers need to be aware that breakdown structures do not refer to the included breakdown elements, but to objects that represent the usage of the respective breakdown element (*BreakdownElementUsageInBreakdown*). Refer to Chap 19.

3.2.2 Breakdown revisions and breakdown element revisions

In S3000L, breakdown revisions and breakdown element revisions manage the changes to breakdowns and breakdown elements respectively.

Each breakdown revision and breakdown element revision define a new release of a breakdown and breakdown element respectively (refer to Fig 4). Revisions trace the development progress (often referred to as development iterations). The use of revisions supports configuration management, and in particular, enables synchronization between changes introduced from Product design, and their impact on the LSA results.

The use of breakdown revisions and breakdown element revisions is especially important for projects where LSA starts during the early phases of the program, and is an integral part of product design and development. Explicit revisions enable an iterative development and allow for explicit traceability between product structures defined as part of product design, and product structures defined for LSA.



ICN-B6865-S3000L0117-001-01

Fig 4 Examples of breakdown revisions defined during the Product development

Note

Adding hardware and software parts to hardware and software elements respectively often results in new hardware and software element revisions.

Note

Breakdown element revisions must not be confused with legacy methods to document alternate solutions within a Product breakdown, such as Alternate Logistics Control Number (ALC) in GEIA-STD-0007, or disassembly code variant in S1000D.

Note

Revision identification can include both release and progress (in-work) identifications. As an example, the S-Series IPS specifications identify the issue number as a value between "000" and "999", in combination with a progress (in-work) identification with a value between "00" and "99". The issue number and progress identification must be separated with a hyphen ("-"), for example "001-01".

3.2.3 Breakdown element identification

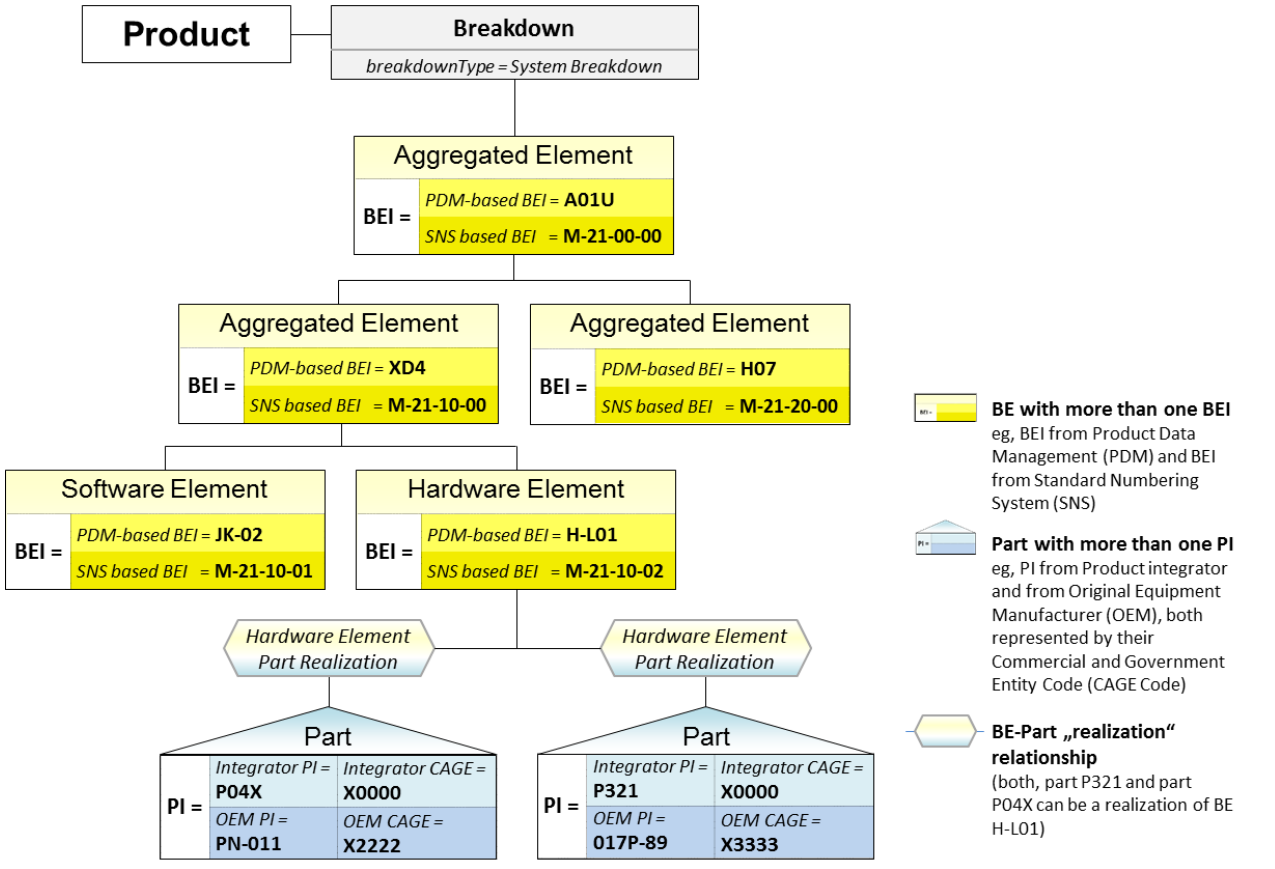
A Breakdown Element Identifier (BEI) is used to identify an individual breakdown element within a Product breakdown structure. A key feature in S3000L is that it allows for more than one BEI to be assigned to one and the same breakdown element (refer to Fig 5).

The rationale for this is that breakdown elements can use identifiers generated by the system, early on in a project, then wait to assign more supportability-oriented identifiers when the Product breakdown structure stabilizes. However, it is possible to present the hierarchical structure using explicit parent-child relationships.

Note

If a breakdown element has multiple BEI, the additional characterizations must differentiate the respective identifier, in terms of identifier type and/or the organization that assigned the identification.

If frequent changes are introduced into the breakdown structures early on in a project, the use of explicit parent-child relationships reduces the amount of work, since it does not require reassignments of BEI due to changes in the design product structure.



ICN-B6865-S3000L0118-001-01

Fig 5 Example of multiple identifiers for breakdown elements

Note

Supportability-oriented BEI provide an explicit understanding of what portion of the Product a given breakdown element belongs to, and of its position in the hierarchical structure. Examples of supportability-oriented identifiers are Logistics Control Number (LCN) in GEIA-0007 and Standard Numbering System (SNS) in S1000D (refer to Para 5.2.9).

Note

If there is a need to establish supportability-oriented breakdown BEI, it is necessary to harmonize them with the other IPS elements, such as technical documentation and material support.

Note

Software implementers must also note that it is possible to assign supportability-oriented BEI to objects representing the usage of breakdown elements in a specific breakdown (known as *BreakdownElementUsageInBreakdown* in the data model, refer to Chap 19 and Fig 11). For example, it is possible to use a breakdown element in different locations in a breakdown, and still differentiate its usages from the breakdown perspective. For example, one breakdown element represents a wheel, which can then be instantiated multiple times in a breakdown structure (eg, right front wheel, left front wheel, etc.).

3.2.4 Breakdown element relationships

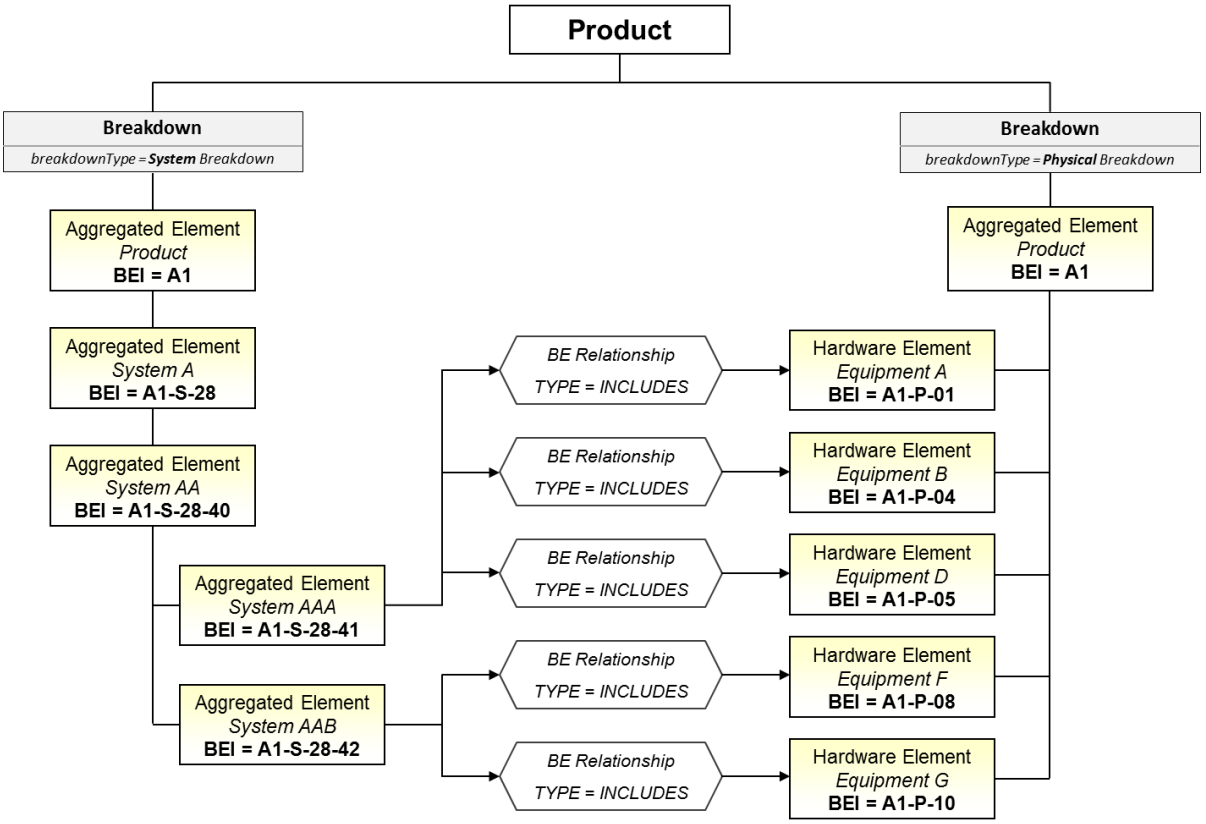
There are two ways to define relationships between elements in breakdown structures.

The first approach defines relationships between breakdown elements (refer to Chap 19, Unit of Functionality (UoF) Breakdown Structure, *BreakdownElementRevisionRelationship*). This approach can be used if the relationship is stable and will not change between different revisions of breakdowns. For example, it is possible to use this approach to:

- include of physical breakdown elements in a system breakdown relationship (refer to Fig 6)
- define the relationship between functional breakdown element and physical breakdown element, like GEIA-STD-0007. Two breakdown elements represent the same thing from different perspectives.
- create an access point relationship, in case one breakdown element is located behind another breakdown element (eg, a panel or hatch)

Note

Using the explicit breakdown element in zone relationship, as defined in the UoF Zone Element (refer to Chap 19), is the most suitable means to define in-zone relationships.

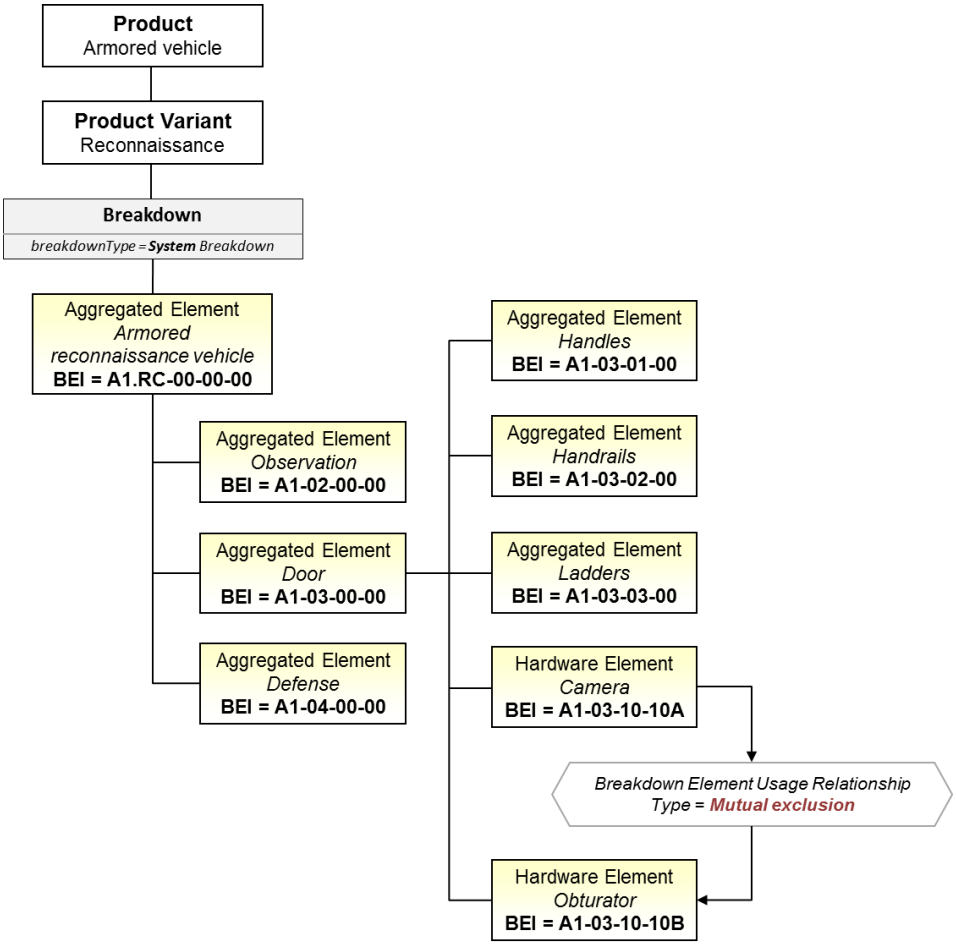


ICN-B6865-S3000L0119-001-01

Fig 6 Example of use of breakdown element relationship

The second approach defines relationships between the different usages of breakdown elements in a given breakdown (refer to Chap 19, UoF Breakdown Structure and *BreakdownElementUsageRelationship*). This approach is most suitable for relationships that represent configuration aspects of breakdown elements in a specific breakdown, such as mutual inclusion and mutual exclusion.

An example of a mutual exclusion is *"If breakdown element X is installed, then breakdown element Y must not be installed"*. For example, these types of definitions are especially useful for mission equipment and mission configurations (refer to example in Fig 7).



ICN-B6865-S3000L0120-001-01

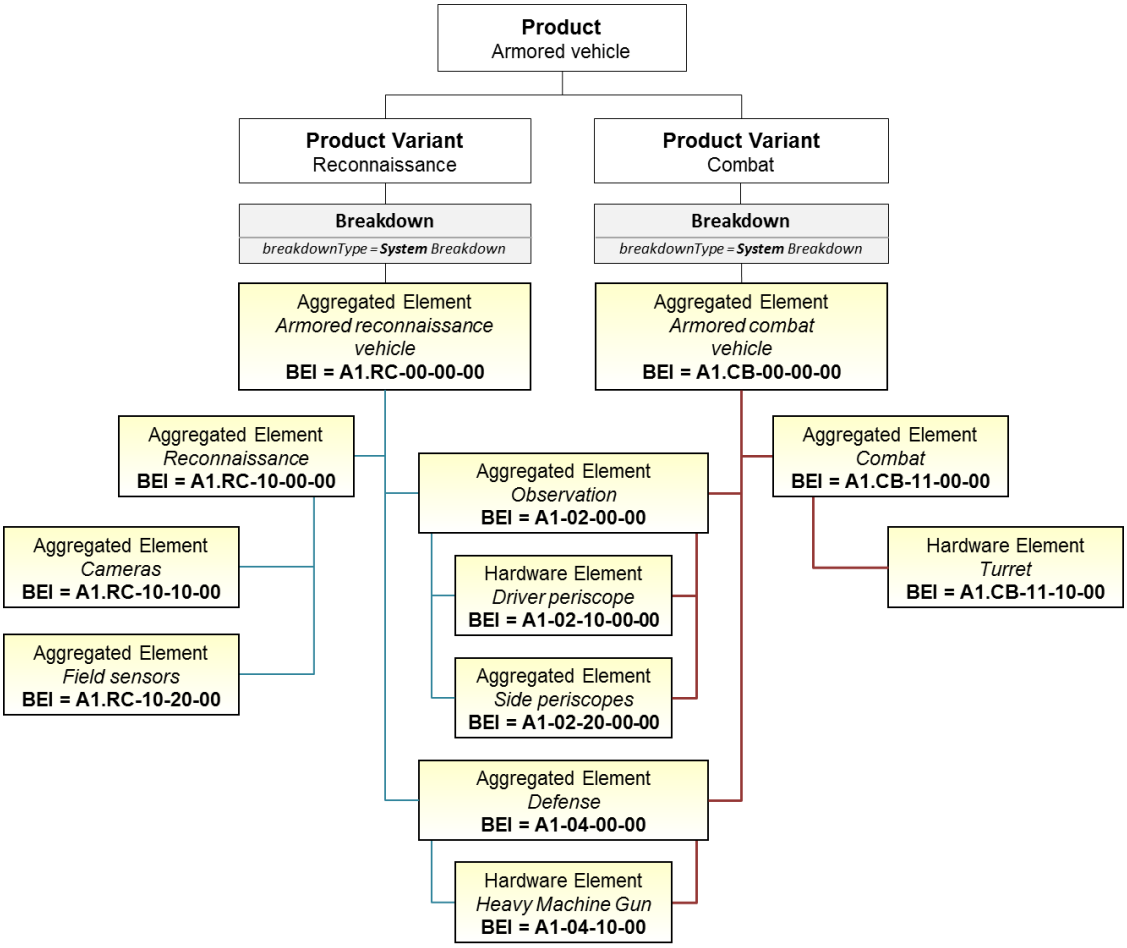
Fig 7 Example of breakdown element usage relationship

Note

The example of breakdown element usage relationship in Fig 7 does not explicitly illustrate the representation of *BreakdownElementUsageInBreakdown* for the respective breakdown element, but rather it conveys a more simplified view (refer to UoF Breakdown Structure, Chap 19).

3.2.5 Breakdown element in multiple breakdowns

As such, breakdown elements are not explicitly included in any given breakdown structure, but their usages are (refer to Chap 19, UoF Breakdown Structure and *BreakdownElementUsageInBreakdown*). This way, it is possible to use the same breakdown element in multiple breakdown structures or in breakdowns defined for totally different purposes (refer to Fig 8). Because this principle is used, it is possible to use the same breakdown element in different breakdown types, therefore there is no need for defining explicit breakdown element relationships between breakdown elements defined in different breakdowns (eg, system versus physical breakdown) for the same Product.



ICN-B6865-S3000L0121-001-01

Fig 8 Example of usage of breakdown elements in multiple breakdowns

3.3 Parts lists

A part list (Bill of Material, BOM) represents the collection of parts included in the assembly of the parent part (refer to Chap 19, UoF Part Definition). A part in S3000L can have zero, one or many associated types of parts lists. In fact, there can be different parts lists for the same part considered from different perspectives. There is often a need to manage both engineering parts lists, often referred to as Engineering Bill of Material (EBOM), and support or service parts lists, often referred to as Service Bill of Material (SBOM).

Examples of the difference between EBOM and SBOM include, but are not limited to:

- an EBOM includes all detail parts needed to manufacture the assembly part, whereas an SBOM only includes the parts that can be replaced in the context of a maintenance task
- an EBOM can also include parts assembled before they are made available as spare parts to the market. This means that the SBOM can include parts which have different part identifiers (part numbers) from those indicated in the EBOM.

Parts included in a part list can also change over time. This can be due, for example, to additional substitute parts added to the EBOM, or to required changes to the SBOM because of changes in the maintenance concept. Therefore, S3000L allows for managing revisions of part lists over time.

S3000L also allows for defining relationships between part list revisions.

Example:

Part list relationship records that the SBOM revision B for part "54555-101" is based on revision C of the EBOM for the same part.

A part can also have one or many alternate parts. An alternate part can replace the base part in all its usages. Therefore, interchangeability is context-independent, and the parts coincide with respect to their form, fit and function.

A part in a specific part list can have a substitute part. If a substitute part can only replace the base part in the context of a specific part list, the interchangeability is context dependent.

Note

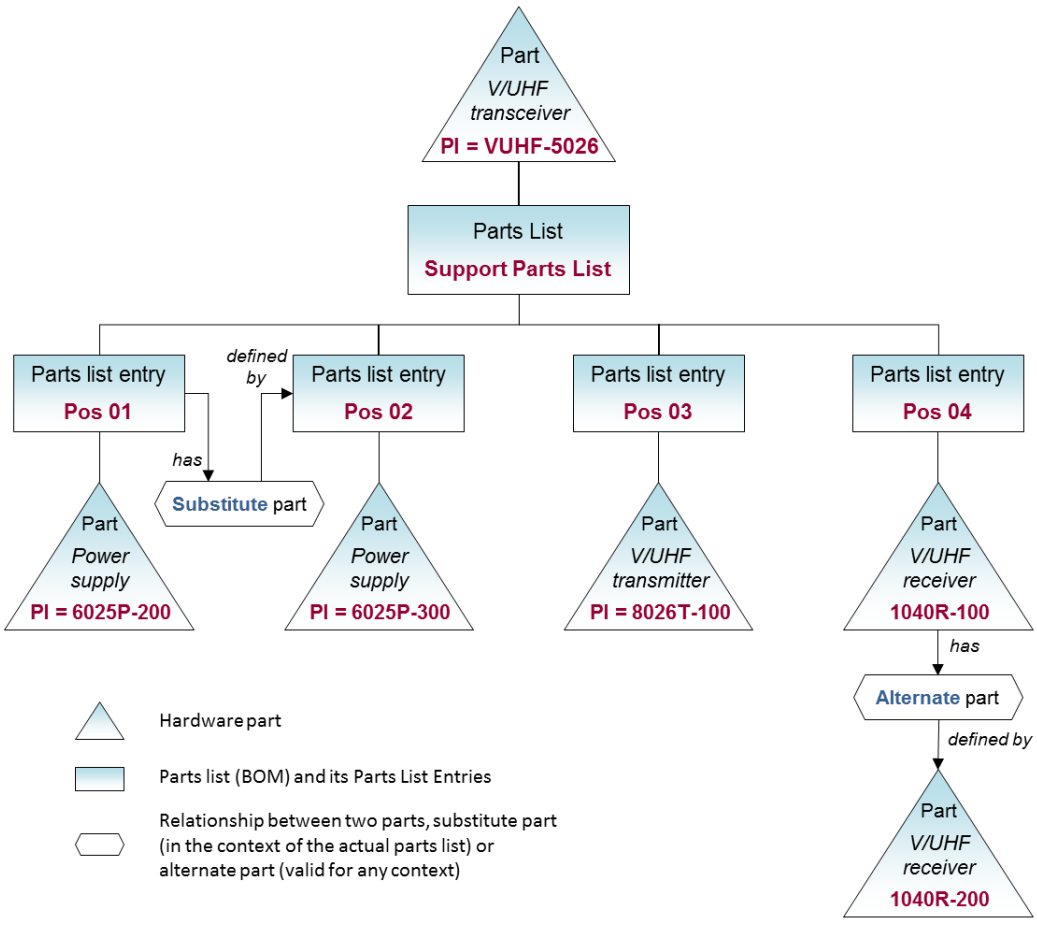
Alternate parts are often defined for details such as nuts and bolts. In most cases, substitute parts are defined for equipment, either as substitute parts within a parts list or as alternative realizations for an installation location on Product level. The latter is best represented using breakdown element realizations in S3000L (refer to Para 3.2.1).

Note

It is not possible to represent alternate or substitute vendor parts using the multi-valued identifier attribute for a *Part As Designed*.

Note

The terminology of alternate and substitute parts is used in most product data standards, such as ISO 10303:239.



ICN-B6865-S3000L0122-001-01

Fig 9 Example of part list and the use of substitute and alternate parts

Fig 9 illustrates that V/UHF transceiver (VUHF-5026) is an assembly of a power supply, a V/UHF transmitter and a V/UHF receiver. The SBOM for the V/UHF transceiver (VUHF-5026) indicates that there are two power supply parts (6025P-200 and 6025P-300), and both can be used as the power supply in the V/UHF transceiver (VUHF-5026). The substitute part relationship specifies that it is not possible to install both at the same time, but that they are interchangeable in the context of the V/UHF transceiver VUHF-5026.

Note

Since the power supply 6025P-300 is defined as a substitute part for power supply 6025P-200 in the context of the parts list for V/UHF transceiver VUHF-5026, this only allows for power supply 6025P-300 to replace power supply 6025P-200 in the context of V/UHF transceiver VUHF-5026, and not in any other context unless explicitly defined.

On the other hand, V/UHF receiver 1040R-200 can replace V/UHF receiver 1040R-100 in any usage, and therefore does not need to be explicitly defined as a substitute in the context of V/UHF transceiver VUHF-5026.

Note

It is necessary to define substitute parts as part list entries in every part list where they can be used as an alternative part.

Note

In the example above, identification of each entry in the part list is unambiguous. In many cases, substitute part list entries are listed with the same position number, often derived from in a drawing. This kind of information can be represented in S3000L using the part list entry attribute "reference designator".

S3000L also considers software as parts. This means that software parts can be included in parts lists for both hardware parts and software parts.

3.4 Replaceability aspects

Replaceability can be defined for hardware elements as well as for parts. In both cases, replaceability can be defined from either a technical perspective or a chosen support strategy perspective. The technical perspective defines whether the associated part can be replaced (yes/no), whereas the replaceability strategy defines, for example at what maintenance level the replacement takes place (refer to Chap 11).

3.5 Product design configuration

Product design configuration defines allowed configuration standards for a given Product or product variant. Product design configuration allows the user to define permitted combinations of breakdown elements, hardware parts, software parts and parts list entries in the context of a Product, of a given product variant, or of a highly configurable assembly part. Allowed product configurations are also often referred to as effectivity definitions.

The UoF Product Design Configuration (refer to Chap 19) can restrict the allowed usage for breakdown elements and parts, included in more generic product structures by restricting their applicability to specific allowed product variants and/or specific product design configurations. It is possible to restrict the respective breakdown element, breakdown element realization and parts list entries to be effective just for a given product variant, a given part-numbered item, and/or for a specific serial number range.

S3000L supports two basic principles for defining product design configurations. These are:

– usable on product variant, where an item (breakdown element, breakdown element realization or parts list entry) in the Product structure for the overall Product is defined as being effective only for a given product variant, or even for a specific serial number range of a product variant

– effective on product configuration, where detailed allowed design configurations can be defined in order to identify allowed configuration (build) standards

Note

The rationale behind including product design configuration information in the context of support analysis is to filter elements in the Product structure to allow the analyst to identify prioritized elements. For example, this allows meeting delivery schedules for specific product variants and/or for a defined Product design configuration (build standard). Second, this information can be useful to the customer/operator to ensure consistent Product configuration and also to filter out task steps and resources needed in performing a task on a certain product variant.

Note

When shared with the customer/operator, LSA must not be considered as an authoritative source for allowed product configuration information.

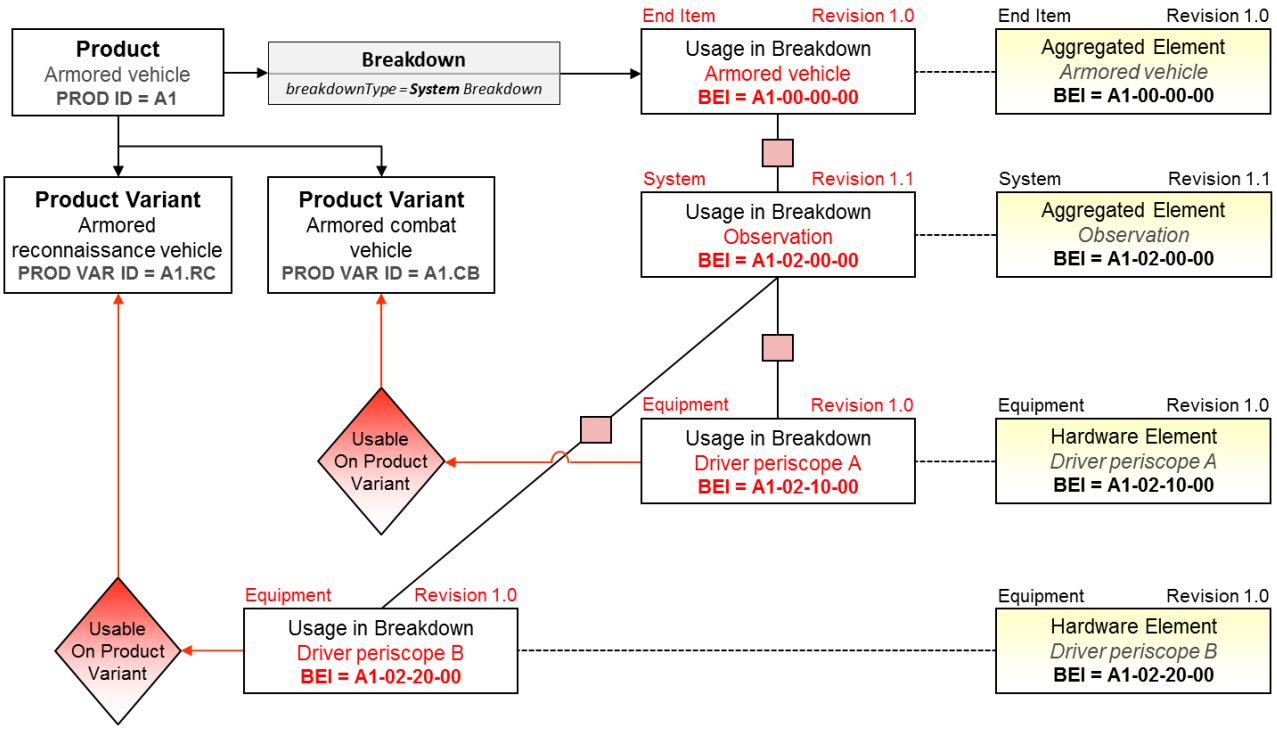
3.5.1 Usable on product variant

If the Product structure contains certain elements which are defined for the overall Product but are only effective for certain product variants, it is necessary to identify those elements by providing *"usable on product variant"* information (refer to Fig 10).

It is also possible to associate usable on product variant information with a limited serial number range.

Additionally, there is also the possibility to apply usable on product variant information at different levels of the Product structure, for example:

– at breakdown element level, usable on product variant can be defined for the breakdown element independent on breakdown type and breakdown revision
– at breakdown element realization level, usable on product variant can identify which part numbered items can be used on specific product variants
– at breakdown element usage level, usable on product variant can be defined per breakdown element in the context of an individual breakdown revision



ICN-B6865-S3000L0123-001-01

Fig 10 Example of usable on product variant construct

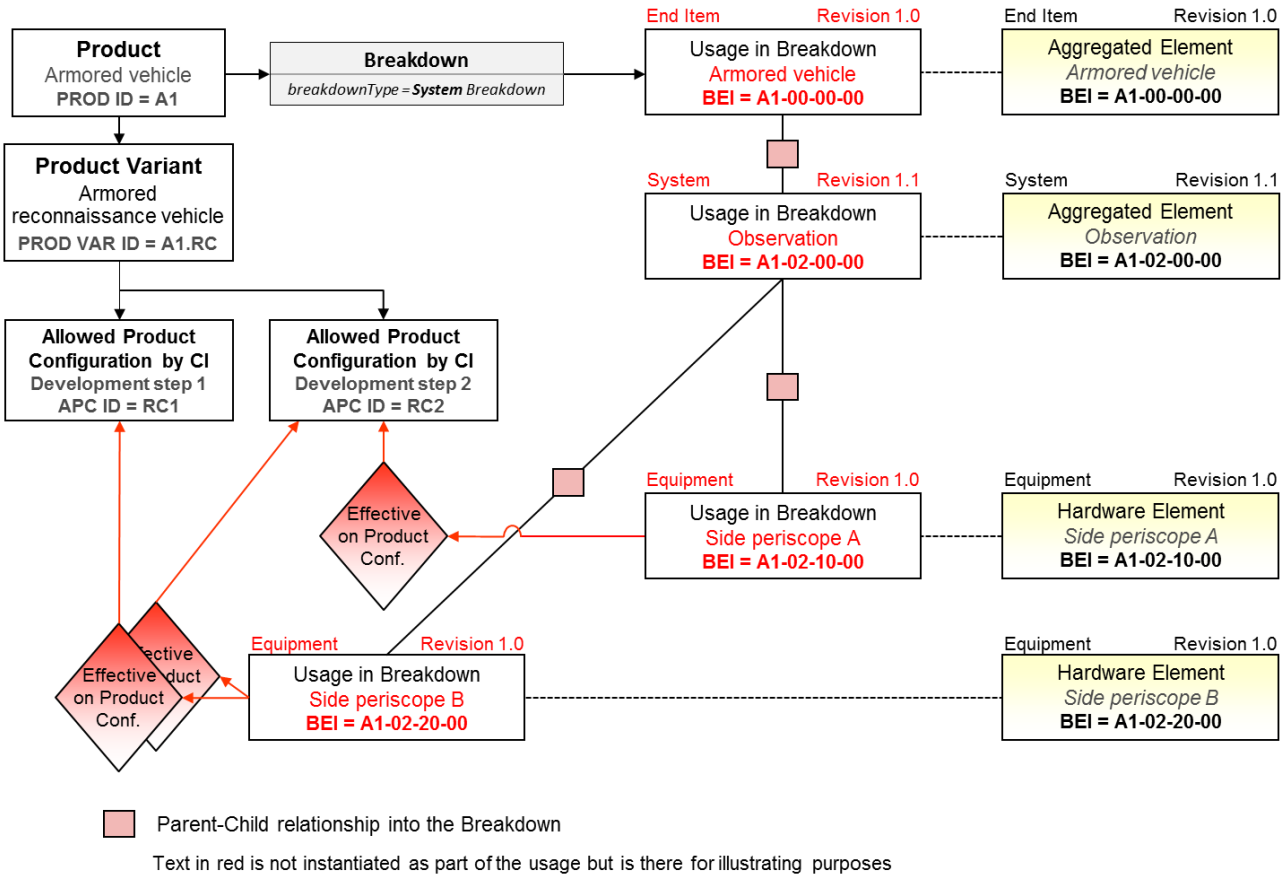
3.5.2 Effective on product configuration

Effective on product configuration information can manage very granular definitions of permitted combinations of breakdown elements, hardware parts, software parts, part list revisions and part list entries, either in the context of a product variant or in the context of a highly configurable assembly part (allowed product configuration). Refer to Fig 11.

There can be a need for an allowed product configuration to have an authority to operate (eg, type certificate), often granted by a government entity.

For example, an allowed product configuration can be used to further define specific allowed configuration standards to meet specified development steps for a given product variant. In order to fulfill the allowed configuration standards, set by the manufacturer, the operator must adhere to the allowed product configuration. The scope of many LSA programs does not include the exchange of this information and does not claim that the LSA data is the authoritative source. However, it can be important in order to:

- identify prioritized elements, for example based on development plans and delivery schedules
- define detailed applicability statements for task-related information



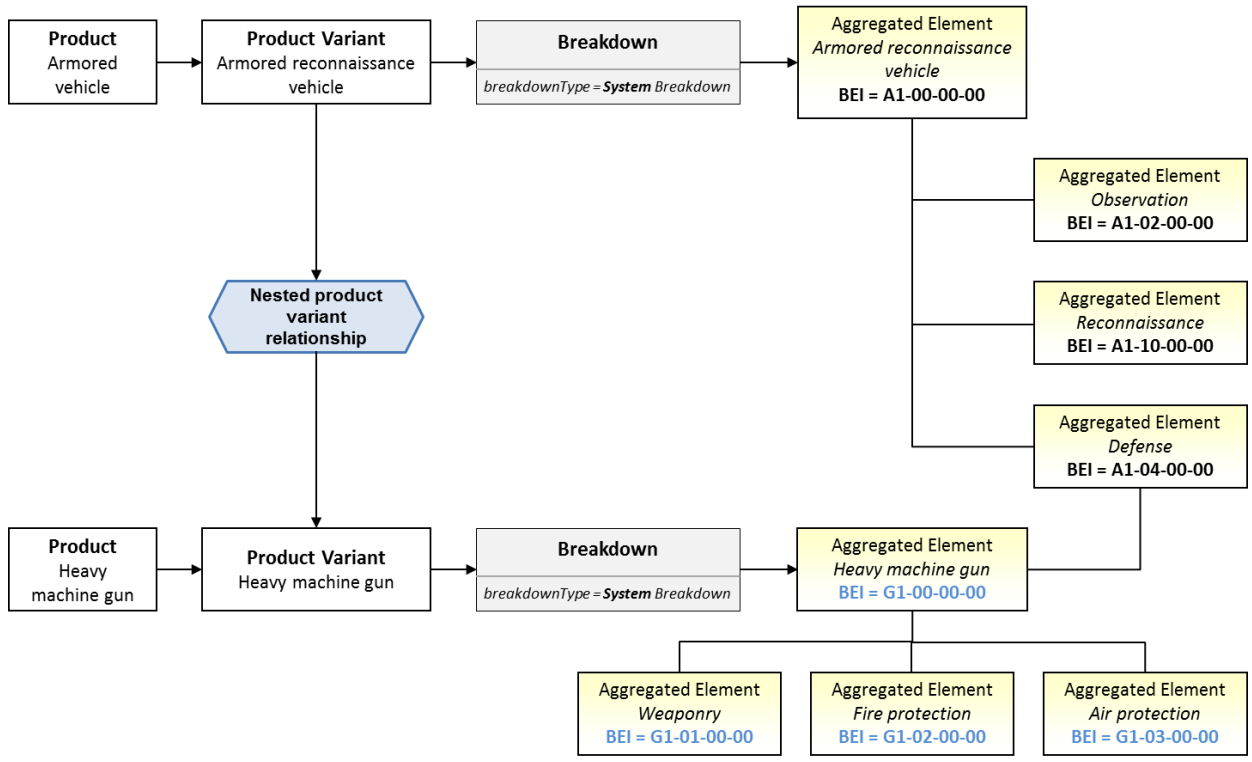
ICN-B6865-S3000L0124-001-01

Fig 11 Use of product configuration

3.6 Nested product variants and product configurations

S3000L has also the capability to define nested structures for product variants and allowed product configurations (refer to Fig 12).

This allows for defining effectivities regarding use of sub-Products in overall Products, for example to define that it is possible to use a specific engine model only on a specific aircraft variant, or that certain configured mission kits are only suitable for certain variants of an armored vehicle.



ICN-B6865-S3000L0125-001-01

Fig 12 Example of nested product variants and their nested breakdowns

Note

Nested product variants are not the same concept as a system of systems. A system of systems defines the aggregation of independent Products that, if considered together, provide an overall operator capability (refer to Para 2.3).

4 Product structures from Product design and development

Product structures coming from product design and development are one of the most important inputs for LSA activities. Design Product structures can come in many different forms and formats, including 3D model representations. This is one of the reasons why S3000L has a highly flexible data model, which can represent many Product structure philosophies using breakdowns and parts lists.

Product design and development activities will often generate more than one Product structure. Typically, there will be at least one system-oriented breakdown and one physical product structure. The representation of a system-oriented breakdown is often a hybrid breakdown which consists of both pure system elements both often refers to physical elements as its leaf elements.

The representation of the physical Product structure can vary depending on the Computer-Aided Design (CAD) tool and/or the Product Data Management (PDM) system being used. There can also be functional or zonal breakdowns. It is necessary to determine the relevant Product structures for the LSA activities as early in the program as possible.

In most cases, design Product structure(s) will change over time, often as a result of continuous Product improvement and/or new Product requirements. It is important that LSA activities track and react to these changes.

The frequency and extent of Product design changes often depends on the maturity of the Product. The earlier the LSA activities are initiated during Product design and development, the more changes LSA activities must be able to manage (refer to Para 6).

It is important that LSA can establish an unambiguous traceability between the design Product structure(s) and the Product structure(s) used as the basis for the LSA activities. The ideal situation is that both Product design and LSA could work against one and the same set of Product structures.

In the case the definition and management of the Product structures for LSA occur in a different environment than Product design and development, it is recommended to start LSA activities by establishing the Product structure that reflects the design Product structure, and keep them separated from the Product structures used for LSA. It will then be easier to identify changes introduced into the Product design in order to analyze their impact on LSA.

Note

S3000L dictates that there must be as least one defined product variant for each Product. It is necessary to take this aspect into consideration also when documenting the design product structure.

Para 3.2 and Para 3.3 describe the means of establishing a design product structures.

5 Product structures for support

5.1 Introduction

Early on in the program (refer to Para 4), it is necessary to determine whether the design Product structure(s) can also be used for LSA, or if there is a need to define new Product structure(s) for LSA.

Either way, there is a set of requirements to consider when defining Product structure(s) that support LSA activities. These are, for example:

- define breakdown elements, which represent families (collections) of parts such as wires, composites, tubes, etc. These are often needed to identify general repair procedures.
- assign operational tasks, as well as functional tests, to relevant levels of systems/subsystems and/or functions/sub-functions
- define breakdown elements for all LSA candidates that will be maintained and/or be replaced on Product, including “LRU in LRU” (LRU = Line Replaceable Unit)
- define zonal location and access points for installation locations
- assign supportability-oriented identifications (refer to Para 5.2.9)

The Product structure(s) defined to support the LSA process and the relevant analysis activities have the following objectives:

- give a clear understanding of how the Product is structured with respect to systems, subsystems, functions, hardware and software. This also applies to enabling peripheral equipment that can be included in the LSA process (eg, complex training equipment and support equipment).
- give a clear relationship between the included hardware elements and their possible realizations in terms of manufactured hardware parts
- give a clear relationship between the included software elements and their possible realizations in terms of actual software releases (software parts)
- enable the allocation of supportability-oriented identifiers
- enable an appropriate level of detail for the selection of LSA candidates
- identify product variants and allowed product configurations
- support configuration management activities with respect to how a specific revision of the Product breakdown relates to the revisions of the corresponding LSA results. It must support configuration control of source data on which the LSA results are based, including documents, design information and Product usage data.

There is often a need to define a set of breakdown structures for the Product, each serving a specific purpose. Many projects will have a design breakdown structure for the Product being defined in a PDM system. The same Product often also has an LSA breakdown, as well as a separate breakdown structure for recording operational and maintenance data feedback. Having different breakdowns for different purposes requires a controlled process to ensure consistency between the different disciplines.

Since the breakdowns and principles depend on the Product or project, S3000L does not dictate any specific approach.

5.2 Breakdown structure for support

When defining breakdown structures for support, it is necessary to identify:

- Products and product variants in scope for the support program
- Product breakdown structures defined during Product design and development, which can be used as sources for defining the Product breakdown for support
- the way to relate/associate Product breakdown structures for support with Product breakdown structures coming from Product design and development (eg, are they defined as extensions or as additional breakdown structures?)
- the types of Product breakdown structures required to support the support analysis activities and the recording of required outcomes from the support analysis activities
- the level of detail of the respective breakdown structure (eg, should a specific breakdown structure include subsystems, LRU, components)

Note

Breakdown structures often contain a mixture of breakdown element types such as system elements, physical elements and even zonal elements (also referred to as hybrid breakdown structures). Regardless of the types of breakdown elements included, it is important to define the purpose for the breakdown using its breakdown type attribute.

5.2.1 Product and product variant

It is necessary to identify all Products and product variants that the LSA analysis activities must cover. This also includes possible sub-products, which are integrated into the Product or are part of a system of systems platform.

Example

Examples of sub-products which are integrated within an overall Product are:

- engine on an aircraft
- mission kit on a combat vehicle

Example

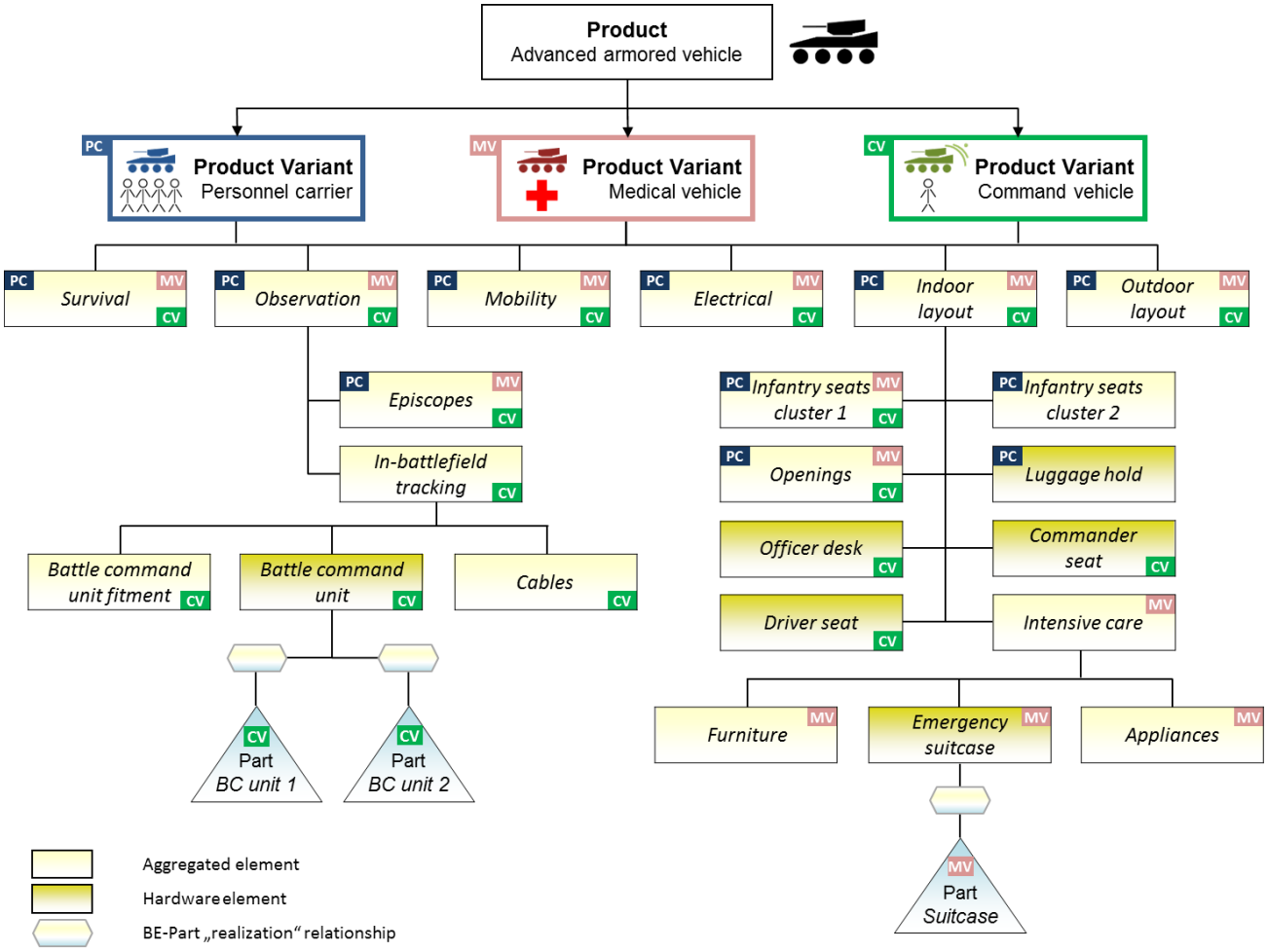
An example of Products, which are part of an overall system of systems platform, is an unmanned aircraft together with its ground station, Ground Support Equipment (GSE) and training simulator.

Each Product can have its own Product structure that is a unique combination of systems, subsystems, component parts/materials and software. Refer to Para 5.2.2.

It is advisable to identify and name Products and product variants early in the program, and this process must preferably involve Product design and development and other IPS disciplines.

Note

A Product is referred to as an End Item Acronym Code (EIAC) in GEIA-0007, and a product variant is typically referenced by a Usable On Code (UOC).



ICN-B6865-S3000L0126-001-01

Fig 13 Example of a Product and its product variants

Fig 13 uses the colored markings in each defined breakdown element to identify the usable on product variant effectivity. For example, the breakdown element for *"Openings"* is effective for all three product variants, while the breakdown element for *"Officer desk"* is only effective for the command vehicle product variant.

5.2.2 System/functional breakdown structure

The terms system and functional breakdown are often used as synonyms to refer to a partitioning of a Product into a set of systems and subsystems from an engineering point of view.

The term system breakdown only indicates this type of breakdown structure. On the other hand, the term functional breakdown refers to the partitioning of functions typically provided by the Product from a user/operator point of view, such as taxing, take off, flying or landing for an aircraft. Each function requires one or many systems to realize the function in the same way, as one system can support one or many functions.

Note

A functional breakdown structure can also reflect Product functions (capabilities) as acceleration, speed control, etc.

A system breakdown will normally have a set of hardware and/or software elements as its leaf nodes (eg, the lowest level of the breakdown structure). Given a defined maintenance concept, it is recommended that the system breakdown required for support analysis activities be

stopped at the level where equipment (assembly parts) will be replaced/repaired directly on the Product. A system breakdown for support analysis should include nested hardware and/or software elements where it is possible to perform maintenance or operational tasks on serialized parts within the equipment when installed on the Product. The rationale for this is to be able to define and describe tasks to be performed on Product as opposed to tasks to be performed on bench. Fig 14 provides a simple system breakdown example.

A system breakdown is often the basis for defining and documenting LSA analysis activities and the resulting task requirements from the perspective of the Product.

Note

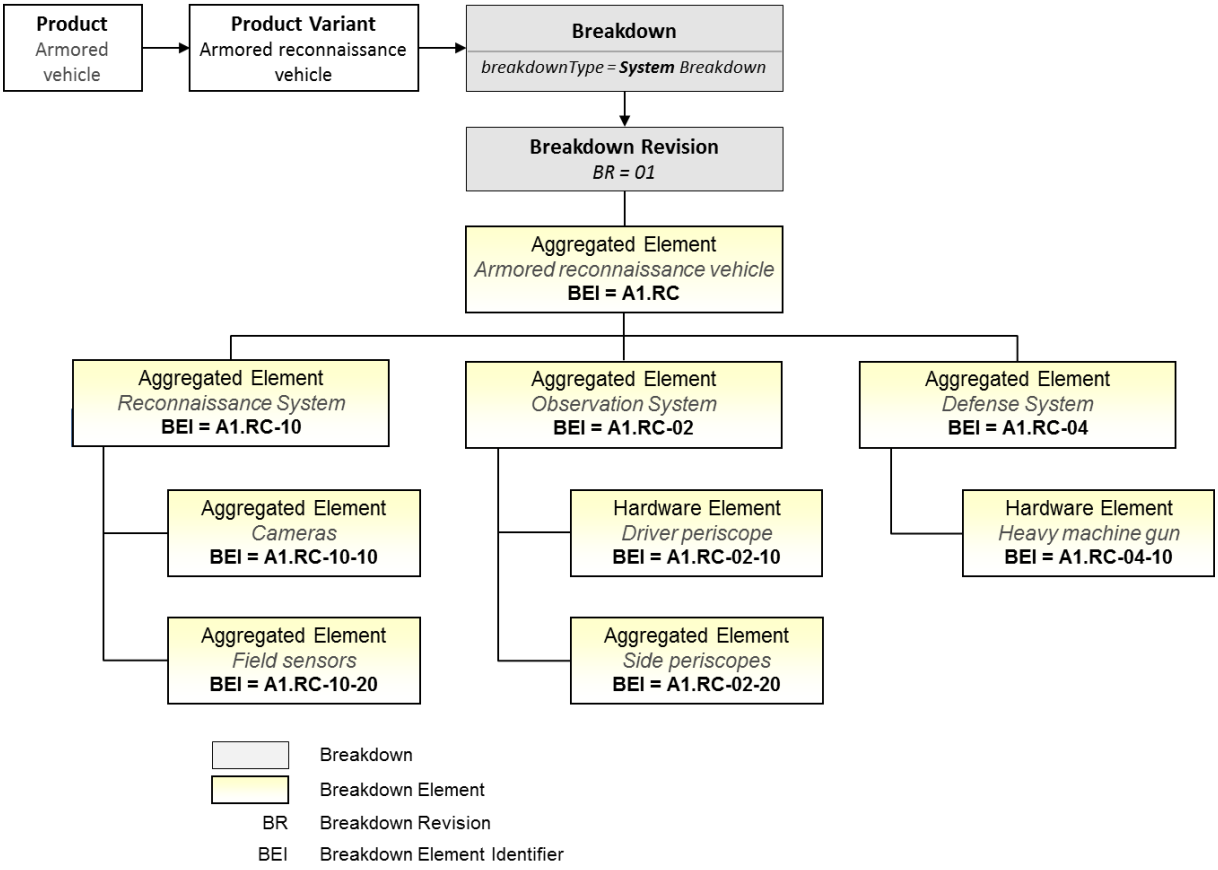
It is also possible to associate LSA analysis activities and task requirements with part definitions (refer to Chap 12).

Note

Although S3000L does not prevent users from defining breakdown elements all the way down to individual piece parts, it is not the recommended practice.

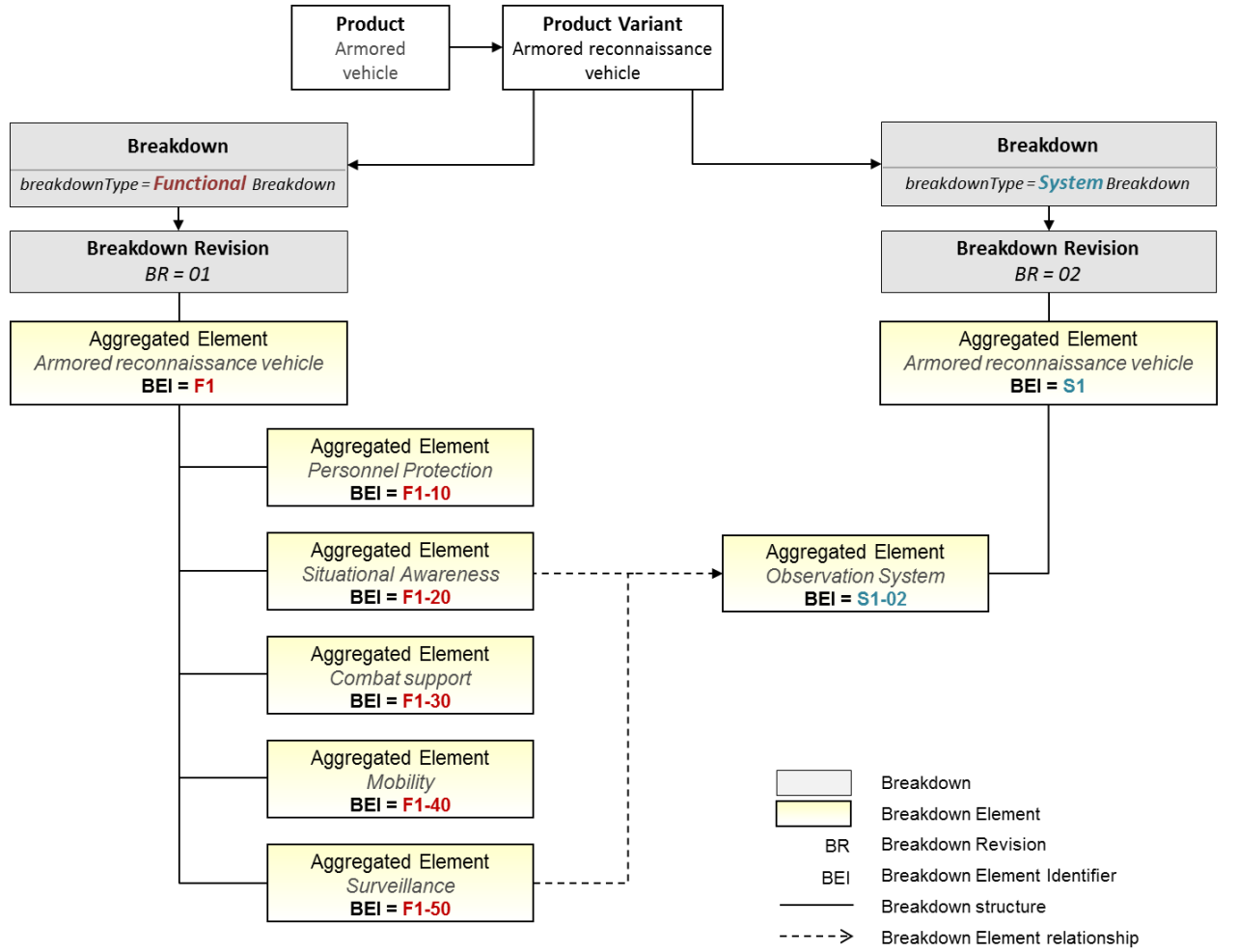
Note

A recommended practice is to define unique hardware and software elements for each installation location within a Product, even where there might be multiple elements that share most of their characteristics (eg, left front wheel and right front wheel). However, this is a decision that needs to be made for each individual program and/or Product. It is also important to distinguish between the part and the usages of the part, where the usage of parts is represented as breakdown elements (eg, a system breakdown structure).



ICN-B6865-S3000L0127-001-01

Fig 14 Example of simple system breakdown



ICN-B6865-S3000L0008-004-01

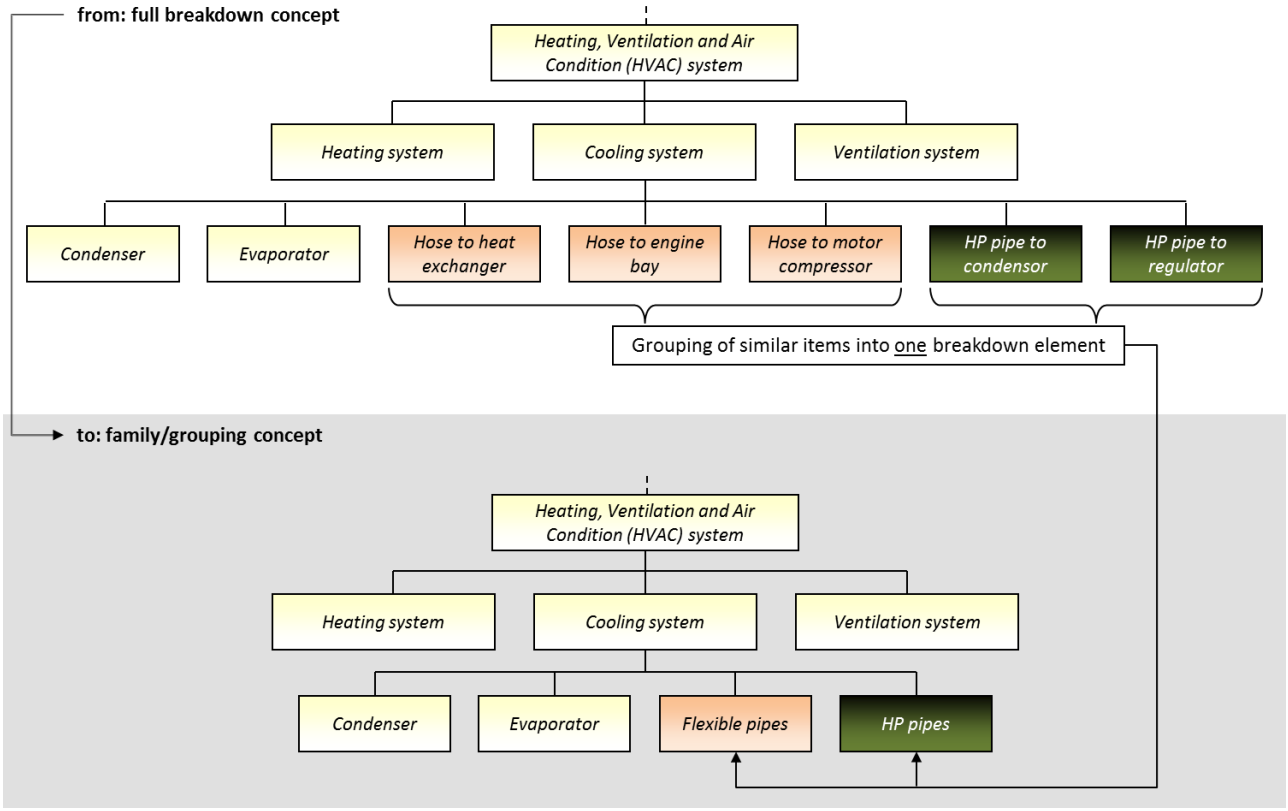
Fig 15 Example of a functional breakdown structure for a Product

Note

Fig 15 shows an example of a simple functional breakdown. The breakdown elements on the left side of the figure represent the functional part, which is connected to a parallel system breakdown via a breakdown element relationship. In this case the *Observation System* S1-02 contributes to provide the two functions *Situational Awareness* F1-20 and *Surveillance* F1-50.

5.2.2.1 Family concepts in system breakdown structures

Similar items can be grouped together for analysis purposes. The family concept can minimize the efforts and avoid the repetition of similar analysis activities. It is the most effective concept for detail parts which share common maintenance/repair procedures.



ICN-B6865-S3000L0013-003-01

Fig 16 Example of family groupings

Note

The example in Fig 16 defines the *"Pipe"* family elements at the *"Cooling system"* level. It is also possible to define a family element at the Product level. For example, it is possible to group all pipes used on the Product and belonging to a specific category, regardless of the system and subsystem they belong to.

Depending on Product breakdown philosophies defined for an individual LSA project, it is possible to include a family breakdown element either in a system breakdown or in a physical breakdown. It is recommended that family breakdown elements be included in the system breakdown, since task requirements are mostly assigned to breakdown elements in the system breakdown rather than to breakdown elements in the physical breakdown. General repair procedures (often referred to as standard repair procedures) are the typical task requirements that can be assigned to family breakdown elements.

Examples of items which can be organized using the family concept:

- wiring of the same type (eg, standard copper, coaxial, fiber-optical)
- structural parts of the same type
- Parts of a similar type coming from one manufacturer. They can be grouped if the maintenance of these parts is covered by a common maintenance concept supported by the manufacturer itself (eg, with corresponding repair kits).

5.2.3 Physical breakdown structures

Physical breakdown structures can be used to define and represent, for example:

- data coming from drawings and CAD systems, including 3D models
- assembly definitions from the Product perspective

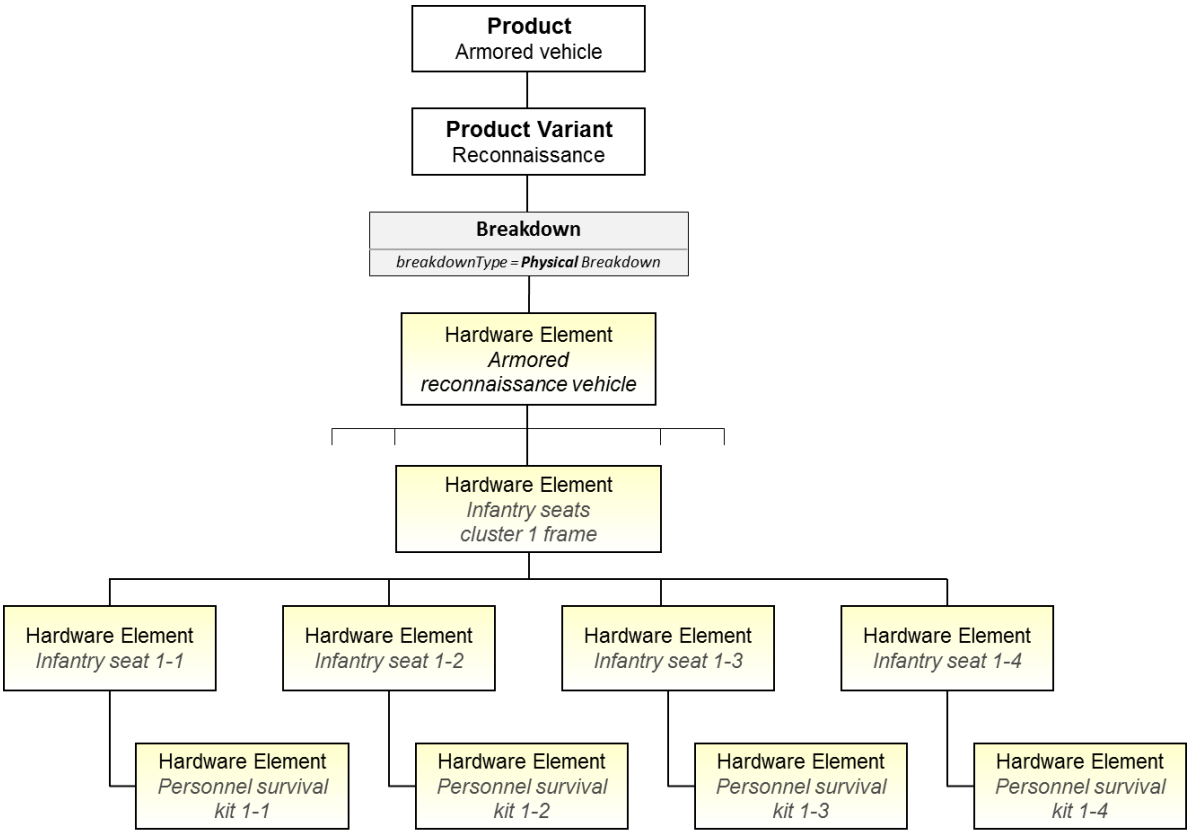
Note

A 3D model approach often results in a concept that identifies positions in the design space using x, y and z-coordinates (sometimes referred to as instances). If there is a need to refer to defined 3D instances as resources for a task, for example, it is possible to transform those coordinates into a physical breakdown structure.

Note

Physical breakdown structures are also in the scope of S2000M provisioning projects and S1000D illustrated parts data. Before defining physical breakdown structures for support analysis activities, it is advisable to consider a joint effort covering all impacted S-Series IPS specifications used in a project.

Physical breakdown structures can also be used to represent assembly relationships for major equipment (eg, a personal survival kit fitted on the seat, refer to Fig 17). This is especially useful where an equipment belonging to different systems has physical dependencies and it can be important to document them as part of remove, install and disassembly tasks. This kind of structure can also be an important input for fleet management and in-service product structures.



ICN-B6865-S3000L0009-003-01

Fig 17 Example of a physical breakdown structure for a Product

In the context of LSA, it is recommended that physical breakdown structures include only physical (hardware) elements down to the level of parts affected by the maintenance tasks at Product level. If possible, these structures must also include attachment parts. The physical breakdown for the overall Product must not include parts which will only be affected by maintenance tasks when removed from the Product (eg, on bench). These parts are just defined in the context of part lists (refer to Para 3.3).

A physical breakdown structure typically starts at Product level, and there are different ways to organize it considering different aspects, for example:

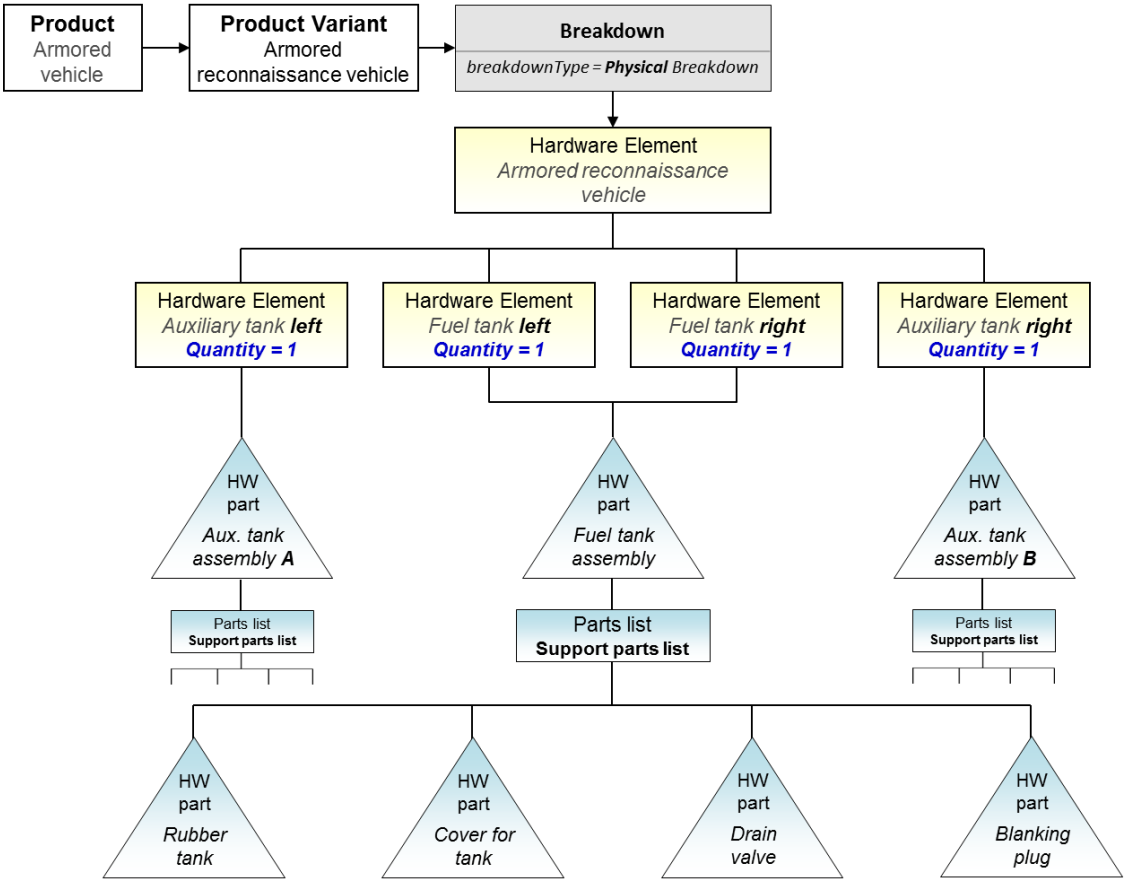
- Product sections (eg, front, middle and aft fuselage)
- Product bays (areas) which organize the physical breakdown in accordance with, for example, access points and installation locations
- major systems
- Access-oriented structures. The access points are defined on the first indenture level, and followed by the installation locations which can be directly accessed. In case of removal of the respective equipment at a given installation location, this will be listed as a child element underneath that installation location.
- equipment assembly dependencies seen from installation locations, for example:

  - Ejection seat assembly is mounted onto the aircraft fuselage
  - Seat is mounted on the ejection seat assembly
  - Beacon and personal survival pack is mounted on the seat

It is also possible to combine a physical breakdown structure with zonal breakdown elements to identify its physical location (eg, product section, bay).

Note

It is important to understand the difference between a hardware element and a part. A hardware element represents the usage of a part in the context of an overall Product, but it does not represent the part in itself (refer to Para 3.2.1).



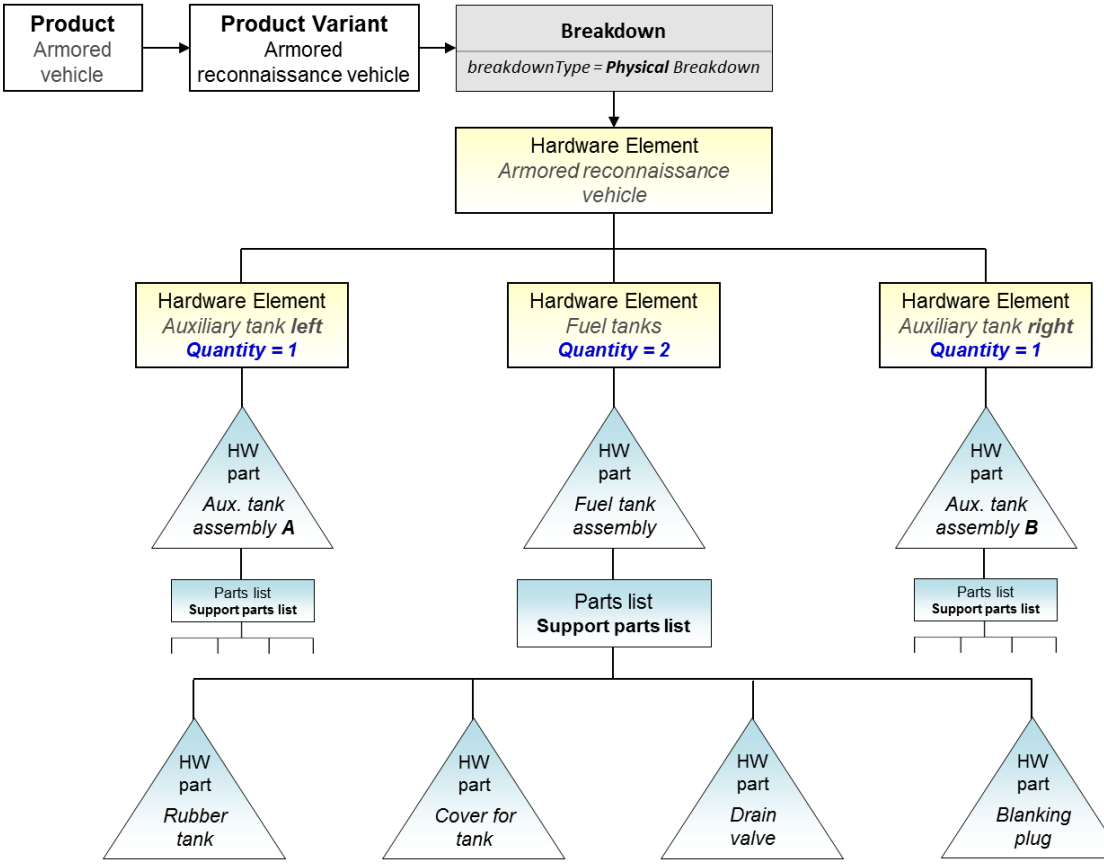
ICN-B6865-S3000L0010-004-01

Fig 18 Breakdown example, all hardware elements with their own representation

In general, the installation location is the primary information within a physical breakdown structure. It is then necessary to define the physical breakdown structure to represent each installation location by individual breakdown elements. The example in Fig 18 illustrates the

situation where two identical fuel tanks are installed on the Product, and each tank is defined as a separate hardware element.

However, it is also possible to define identical physical elements as one single breakdown element with quantity information (refer to Fig 19). In the example below, there are two identical fuel tanks installed within the Product. In some cases, it can be sufficient to have only one breakdown element representing a set of similar/equivalent capabilities.



ICN-B6865-S3000L0011-003-01

Fig 19 Breakdown example, grouping of hardware elements into one single representation

It is recommended to stop the definition of a physical breakdown structure at the level where parts will not be defined as either:

- the target for a task/subtask
- required resources for any on-product maintenance tasks

The physical breakdown structure should focus on defining breakdown elements for:

- the LRU, including LRU in LRU (at any indenture level)
- attaching parts
- parts that need to be removed in order to gain access

The rationale for this is that there is no need to define breakdown elements for parts that are just needed for maintenance tasks on bench, since they are already defined as part of the part list for the assembly. The location of the assembly installation has no relevance for the tasks on bench or for the resources needed. For more information on how to determine whether tasks must be associated with breakdown elements or with parts, refer to Chap 12.

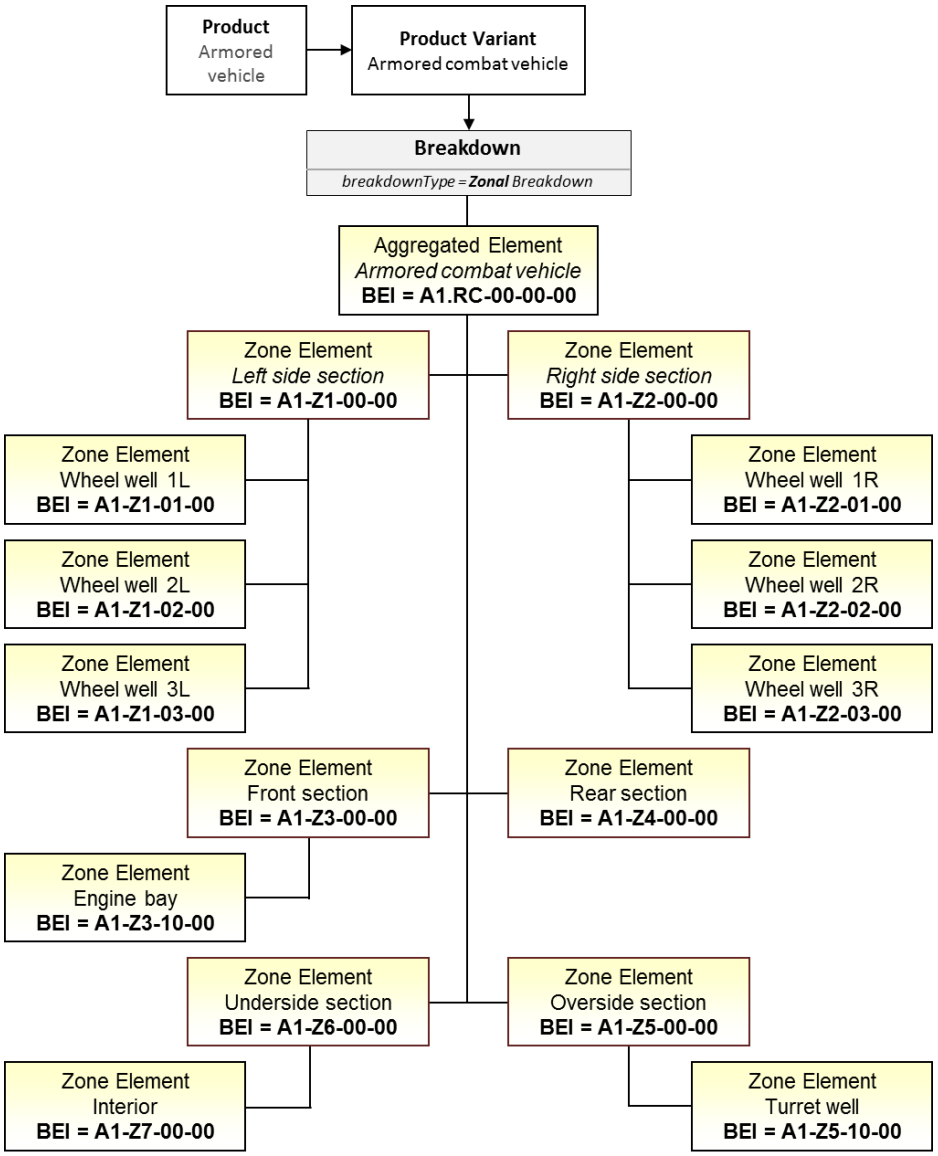
5.2.4 Zonal breakdown structures

The easiest form of zonal breakdown structures is just a list of all zones defined for the Product (or for the defined product variant), without any hierarchical structure. The required depth and structure of a zonal breakdown established in an LSA can be different from those established for design or production purposes. Nevertheless, it is recommended to harmonize zonal breakdown rules within a project. Typically, it is necessary to use zones as breakdown elements to document maintenance activities related to the physical area represented by the zone element (eg, scheduled zonal inspection). Refer to Fig 20.

It is possible to use zones to define the zonal location for hardware elements by establishing relationships between the respective hardware element and its associated zone element.

Among other things, zonal breakdown structures can also be used to determine:

- work areas within a ship
- dangerous zones
- radioactive zones



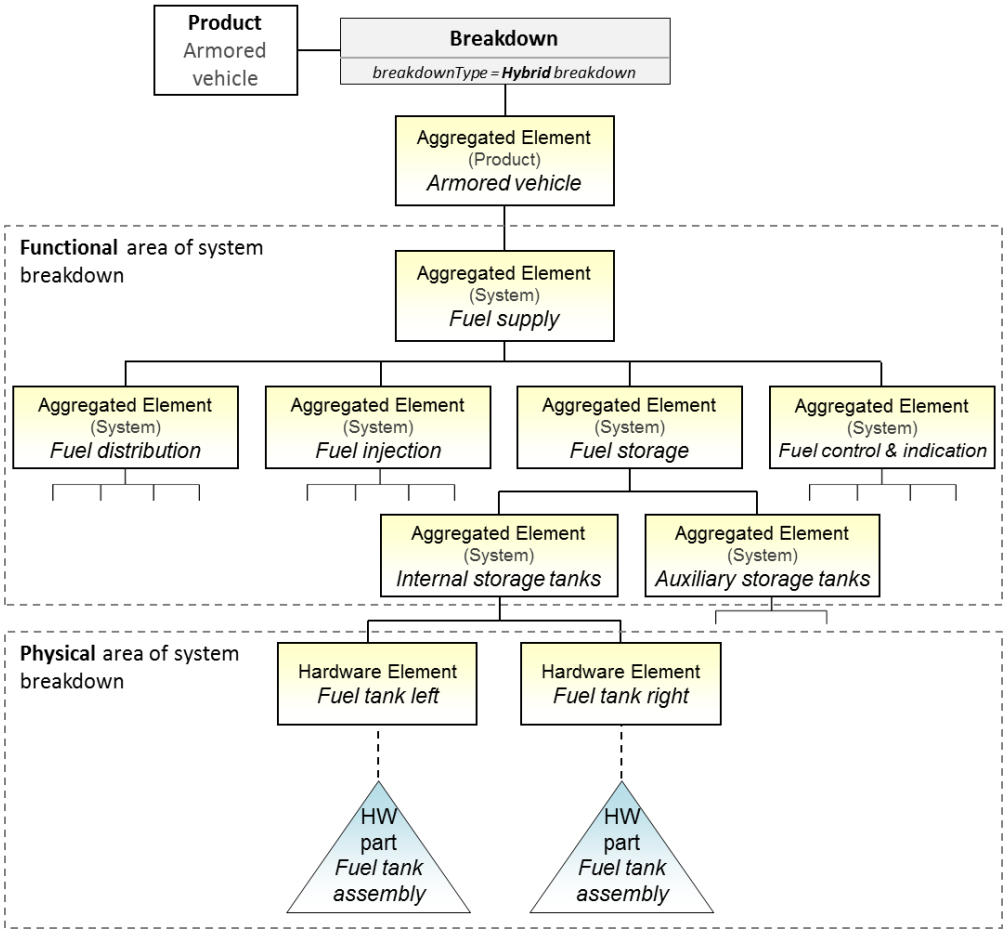
ICN-B6865-S3000L0128-001-01

Fig 20 Example of a zonal breakdown structure for a Product

5.2.5 Mixture of breakdown philosophies (hybrid breakdown structures)

In many cases, Product breakdown structures are a mixture of systems, functions, zones and/or physical aspects, as shown in Fig 21. This type of breakdown is often referred to as hybrid breakdown. For example, hybrid breakdowns subdivide:

- functions into systems and/or physical elements which support the respective function
- systems into functions and or physical elements which realize the respective system



ICN-B6865-S3000L0012-004-01

Fig 21 Example of hybrid breakdown as a mixture of system and physical elements

5.2.6 Interrelating elements in different breakdown structures

If an LSA project requires multiple breakdown structures in order to document different aspects of the Product(s) under analysis, then it is also necessary to determine whether there is an interrelation between breakdown elements in the respective breakdown structure, and how the elements interrelate.

For example, the interrelation of functional elements in a functional breakdown and elements in a physical breakdown can be a good basis for troubleshooting analysis.

Based on a functional breakdown, troubleshooting must be able to identify potential failures that led to the unavailability of certain functions. For this reason, the analyst must know which hardware elements can be responsible for the missing function. For example, the function "*fuel distribution*" can contain typical elements such as fuel pipes, but can also be dependent on electric power supply elements. Therefore, even if the electric power supply is documented in another area, it must be associated with the function "*fuel distribution*".

There are two ways of interrelating elements in different breakdown structures:

- defining explicit relationships between breakdown elements (refer to Para 3.2.4)
- using the same breakdown element in multiple breakdowns (refer to Para 3.2.5)

5.2.7 Managing revisions of breakdown structures and its elements

In S3000L, breakdown revisions and breakdown element revisions manage the changes to breakdowns and breakdown elements respectively. Revisions trace the development progress (often referred to as development iterations). Refer to Para 3.2.2.

Note

Traditional LSA breakdown methodologies do not take into consideration revisions of breakdowns and breakdown elements, which are being introduced in S3000L to improve integration with Product design and other IPS disciplines.

5.2.8 Managing Product design configuration information

The need for managing Product design configuration information from an LSA perspective mostly derives from the need to define the Product effectivity for different equipment at specific installation locations within a Product or product variant (refer to Para 3.5.2). Product design configurations are needed to distinguish between:

- similar equipment from different manufacturers (equal in form, fit, and function) which are allowed in different contexts, for example based on customer preferences (eg, allowed in different product variants)
- modified or upgraded equipment (equal in form, fit, but not necessarily in function) which are allowed in different contexts (eg, serial number ranges for a given product variant)

5.2.9 Supportability-oriented breakdown element identifiers

A BEI uniquely identifies a specific breakdown element. It is possible to define breakdown structures either using explicit parent-child relationships (PDM approach), or letting a supportability-oriented BEI imply the position of a given breakdown element in the breakdown, and how the breakdown element relates to breakdown elements at lower and/or higher indenture levels (traditional LSA approach). Refer to Para 3.2.1.

5.2.9.1 Supportability-oriented breakdown element identifier syntax

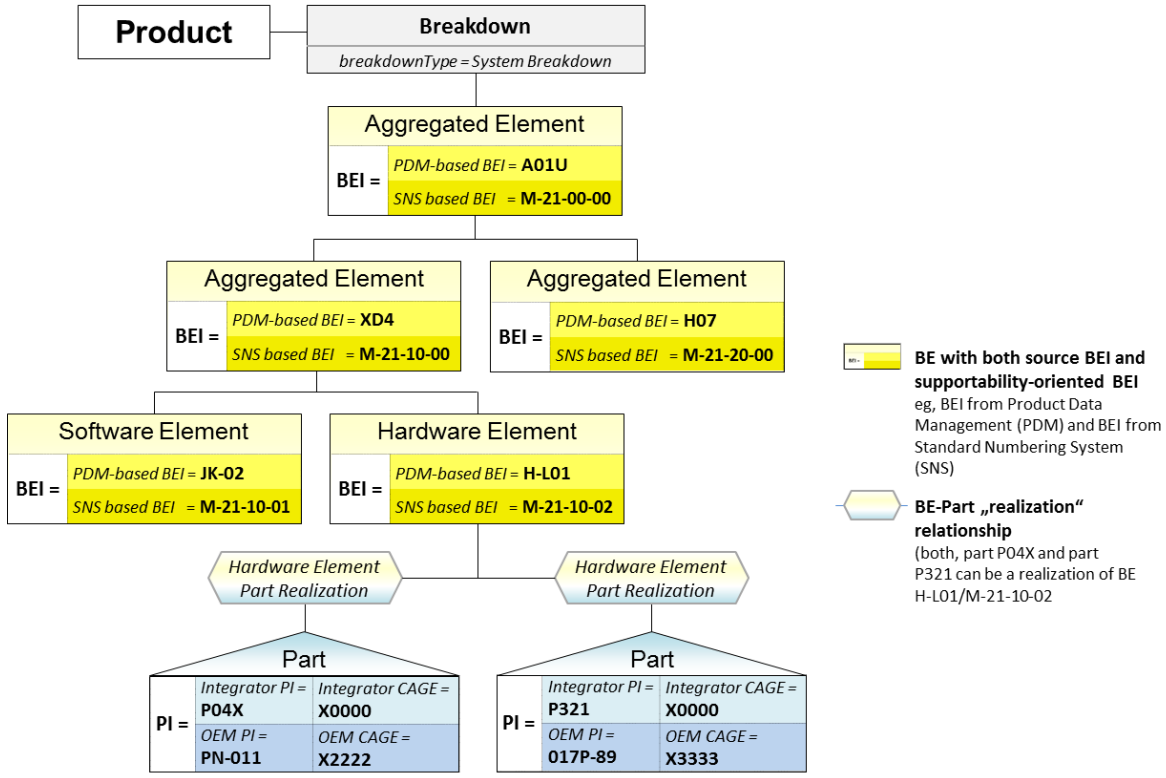
For a traditional Product breakdown approach, the supportability-oriented BEI syntax defines the relationship between the different breakdown indenture levels. It is necessary to define in advance the number and the type of characters assigned to each indenture level.

If the supportability-oriented BEI syntax must also be the basis for defining and managing breakdown structures, it is necessary to define it as early as possible, ensuring that the syntax is appropriate throughout the Product life cycle. In order to use supportability-oriented BEI as the basis for breakdown structure organization, it is necessary to ensure that the BEI syntax is capable, consistent, and broad enough to address the entire project structure, and to deal with all changes that take place, especially in the early development phases.

Note

For projects using the traditional LSA Product breakdown approach, any change to the supportability-related BEI syntax can cause extended rework within different LSA IT systems and associated IPS disciplines. For example, there can be a risk that an indenture level requires more digits than originally planned for, or that it is necessary to reorganize supportability-related BEI due to changes in the breakdown structures.

For breakdown structures using a PDM approach (eg, using explicit parent-child relationships), defining a specific supportability-oriented BEI structure is not required, but it can be assigned when the Product breakdown stabilizes, and subsequent changes will not impact the breakdown elements, nor the associated BEI. Refer to Fig 22.



ICN-B6865-S3000L0129-001-01

Fig 22 Supportability oriented BEI

5.2.9.2 Alternate breakdown element

Projects using traditional LSA Product breakdown approaches will meet the requirement to be able to define alternate breakdown elements like the former usage of ALC from GEIA-STD-0007 or disassembly code variant as defined in S1000D.

Since breakdowns in GEIA-STD-0007 require one breakdown element per combination of function being performed and part that can realize the function there is a need to introduce so called alternate breakdown elements (LCN/ALC in GEIA-STD-0007). An alternate breakdown element within GEIA-STD-0007 uses the same LCN, but different ALC to document alternate breakdown element realizations. The same LCN indicates that the alternate breakdown element occupies the same installation location in the Product breakdown.

S3000L does not have an explicit concept for dealing with traditional alternate breakdown elements but supports the requirement in a different way, primarily by allowing for hardware and software elements to have multiple part realizations and to allow for usable on statements to be defined for individual breakdown element realizations and not just for breakdown elements (refer to Para 5.2.8). An explicit breakdown element relationship which identifies the alternate aspect can also be defined.

Note

If legacy LSA programs using MIL-STD 1388-2B or GEIA-STD-0007 is migrated into S3000L then the LCN and the ALC must both be concatenated into one single BEI in S3000L. Also, LCN type and EIAC could be included in the concatenated BEI in order to uniquely identify its origin.

5.2.9.3 Identification of enabling Products

Besides the Product that is the primary subject of the LSA project, LSA activities can also be required for enabling Products such as:

- support equipment (eg, test equipment, working platforms)
- training equipment (eg, simulators, training rigs)

Those enabling Products must also be part of a breakdown structure from a system of systems perspective.

It is possible to define breakdown structures for enabling Products as separate breakdown structures per enabling Product, where the respective enabling Product is managed as a Product in its own right. They can also be included as breakdown elements in the breakdown for the primary Product.

If enabling Products are managed as a Product in its own right, they can have supportability-oriented BEI, which are totally independent from the supportability-oriented BEI defined for the primary Product.

5.2.9.4 Integration requirements

It must be ensured that each supportability discipline, both on customer and contractor side, identify and reference each breakdown element unambiguously. It is necessary to determine on a project-by-project basis whether this is accomplished using supportability-oriented BEI or by some other BEI.

5.2.10 Hardware and software element characterization

5.2.10.1 Hardware element replaceability and reparability

It is possible to classify hardware elements based on the aspects of replaceability and reparability. These classifications are an important input to determine the support concept for the respective breakdown element (eg, by Level of Repair Analysis (LORA), refer to Chap 11). Reparability and replaceability classifications include discard information, as well as accessibility information. Table 2 and Table 3 define a number of general terms which can serve as a guideline to create the corresponding classifications within a project.

Note

During the LSA Guidance Conference (LSA GC), it is recommended to define a common understanding of typical terms to be used as part of LSA (refer to Chap 3).

Table 2 Aspect of replaceability

Terminology	Description
Directly replaceable	Item that is directly replaceable at the Product level.
Not directly replaceable	Item that is not directly replaceable at the Product level (installed in a parent assembly/equipment). Before replacement, it is necessary to remove the parent assembly/equipment. For example, it is possible to replace the compressor of an engine only after the engine removal.
Non replaceable	Item that is intrinsically tied to another and it is not possible to replace it separately.

Table 3 Aspects of reparability

Terminology	Description
Always repairable	Item that always can be restored to a functional state.
Not always repairable	Item that can be restored to a functional state depending on the failure.
Non repairable (discard part)	Item that is not repairable.

Normally, the reparability information is related to a part and will be documented against the part. It is possible to define reparability against the hardware element in the early phases of an LSA program when the exact part realizations are not yet determined.

5.2.10.2 Software element type

It is possible to classify software elements based on their installation and execution in relation to the Product. This classification is an important input to determine the support concept for the respective software element.

Table 4 includes the definition of some general terms which serve as a guideline for software classification within a project.

Table 4 Aspects of software installation and execution

Terminology	Description
Loadable	Software or data that can be loaded on the Product without removal of the target hardware from the aircraft.
Embedded	Software written to control machines or devices that are not typically conceived as computers.
Distributed	Software that runs on multiple computers within a network at the same time and can be stored on servers or with cloud computing.

6 Change management in LSA

6.1 Introduction

Change management is the process that controls the identification and implementation of required changes within a Product.

Change management from the LSA perspective describes:

- the types of changes that affect previous LSA decisions and analysis results
- how S3000L supports the change management process
- the possible impact of a change on LSA, and how to implement changes

Changes can either be changes identified upstream as part of product design and development, or changes that only have an impact on the support solution, not on product design. Each change can then have an impact on other IPS elements such as provisioning, training and/or technical publications. Therefore, change management must include all engineering disciplines rather than focusing on a single discipline.

6.2 Reason for change

6.2.1 Design change

The introduction of changes to the Product design can occur for several reasons including, but not limited to:

- new or revised capabilities, including revised customer requirements
- new or improved hardware
- new technologies, including new test capabilities
- hardware and software obsolescence issues

Regardless of the type and rationale for the change, it is always necessary to evaluate the change from an LSA point of view. There are also many cases where design changes will not impact the LSA.

6.2.2 Design concessions and waivers

If the existing and approved Product configurations do not comply with the defined requirements, there can be a concession. A concession is granted before the execution of a contract and is an authorization to depart from a particular performance of the contract, specification or reference document.

A waiver is another type of deviation from the defined requirements. A waiver is granted after the execution of a contract, and is an authorization to depart from a particular performance of the contract, specification or reference document.

Both concessions and waivers restrict the specified capabilities of the Product, and often result in restrictions on the Product usage. They can also result in additional or different maintenance tasks (eg, additional and/or more frequent inspections, or reduced authorized life for hardware). Waivers often result in unplanned design changes, while concessions are known and managed as part of the design, as well as during LSA performance.

It is necessary to incorporate into the LSA the waivers which require changes in the support solution, and the resulting changes must refer to change authorizations defined for the waiver.

Changes in the support concept and support solution due to waivers will often be removed once a redesign resolves the reason for the waiver, or when additional investigations show otherwise.

Note

Waivers and their associated changes in the support solution are often communicated as Service Bulletins (SB) to customers and operators.

6.2.3 Preventive maintenance change

Changes in the preventive maintenance defined for the Product (refer to S4000P) can also impact the LSA. By means of an example, the reasons for changes in preventive maintenance can be:

- new built-in test capabilities
- In-Service Maintenance Optimization (ISMO), refer to S4000P

6.2.4 LSA change

LSA change refers to changes which only affect existing results from the LSA activities. This means that there is no change to the Product design itself. For example, LSA changes refer to:

- new/revised requirements for the support concept and/or the support solution
- new/revised regulations that affect the defined support resources
- improvements in the support analysis results based on in-service feedback (refer to Chap 17)
- supplier changes which affect the defined support resources
- new technology which changes the way work can be performed

6.3 Managing changes

The scope of S3000L does not include the definition and description of change management performance. However, it is important to include LSA in the change management process to identify the impact on LSA analysis results, and to ensure there is always a Product support solution which meets the requirements in terms of availability and cost.

It is also important that any change introduced to the LSA results refers to a justification for the change. In most cases, it will be a reference to a change authorization (change order/request). Depending on the scope of the change, it is possible to introduce change authorizations either by design or by LSA.

It is possible to track the impact of a change in the LSA analysis results, including changes in the Product structure for support, using the revision capabilities associated with, for example:

- breakdown revision
- breakdown element revision
- parts list revision
- analysis activity revision
- task requirement revision
- task revision

Objects with explicit revisions are often associated with status information (eg, in-work, released, approved).

It can be necessary to maintain multiple in-service Product configurations or tasks, for example, because they need to exist simultaneously as the result of a change. If this is the case, it is recommended that new objects rather than new revisions be defined. For example, it is recommended to define an additional task, and distinguish the use of the respective task with an explicit applicability statement. It is possible to use applicability statements to define, for example, that task A is applicable to product variant A, while task B is applicable to product variant B.

Note

In the information structure for S3000L, it is possible to define applicability statements at many different levels (refer to Chap 19). The definition of the methods to use applicability statements and revisions must occur as early as possible in the IPS program.

A new revision must override (replace) previous revisions by default. However, it is recommended to keep all revisions and not overwrite them with the latest version (data repository approach).

6.4 Implementation of changes in LSA

6.4.1 Impact of changes to the LSA

For proper change management, it is necessary to correctly evaluate all impacts that changes can introduce into LSA based on the respective LSA activity to be iterated/re-evaluated. At the end, each change can have an impact on:

- the overall support concept, including which tasks are needed (task requirements) and where they must be performed
- how tasks must be performed (steps)
- the resources required when performing a task in terms of material (spares, consumables and support equipment), personnel and infrastructure

6.4.2 Traceability between the source of changes and consequential LSA changes

It is necessary to ensure traceability between the source change and LSA changes.

However, the scope of S3000L does not include a detailed method on the performance of change management.

6.4.3 Change process

Once it is confirmed that a change has an impact on the existing LSA results, it is necessary to perform consequential activities to ensure:

- the correct identification of the impact on LSA
- the identification, assignment and timely planning of the different jobs to implement the necessary changes, and the identification of the necessary resources to implement the change
- the status of the jobs performed by LSA can be known and can be fed back to the overall change management process

It is necessary to record any change which does not affect LSA needs and provide a justification. This shows that the entire change has undergone analysis, and it guarantees the traceability of changes.

6.4.4 LSA update

Update to the LSA results due to a defined change means that any change in the resulting LSA data must refer back to the change authorization.

The process of introducing a change to the LSA must follow the same principles as defined for the respective LSA activity described in chapter 5 to chapter 16. Para 3 illustrates how to manage changes that affect the Product structure.

Chap 20 describes how S3000L can be used to update messages to exchange the result of an LSA update due to change.

6.5 Introducing design changes

The incorporation of design changes (modifications) into a Product in service usually requires maintenance instructions specific for the activity. Those tasks are often included in service bulletins sent to customers and operators. However, it is necessary to analyze and document those tasks in the same way as any other task requirement (eg, documented as part of the LSA).

Design change task requirements should therefore be defined and documented in accordance with Chap 12. The trigger (event driven time limit) for such task requirement/task would be the change authorization (refer to Chap 19, UoF Time Limit).

7 Associated parts of the S3000L data model

The documentation of the associated data to this chapter is supported by the following UoF, refer to Chap 19:

- S3000L UoF Product and Project
- S3000L UoF Breakdown Structure
- S3000L UoF Part Definition
- S3000L UoF Hardware Element
- S3000L UoF Software Element
- S3000L UoF Zone Element
- S3000L UoF Aggregated Element
- S3000L UoF Product Design Configuration
- S3000L UoF Change Information

Chapter 5

Influence on design

Table of contents

List of tables

List of figures

References

Table 1 References

Chap No./Document No.	Title
Chap 6	Human factors
Chap 7	Corrective maintenance analysis
Chap 8	Special event and damage analysis
Chap 11	Level of repair analysis
Chap 12	Task requirements and maintenance task analysis
Chap 13	Software support analysis
Chap 14	Life cycle cost considerations
Chap 15	Obsolescence analysis
Chap 16	Disposal
Chap 17	In-service LSA
Chap 19	Data model

1 General

1.1 Introduction

Influence on the Product design (hardware or software) is one of the main objectives for staff involved in supportability engineering and in the LSA process. Therefore, the Product design program must consider results of the LSA program from a supportability perspective.

1.2 Purpose

The objectives of LSA are:

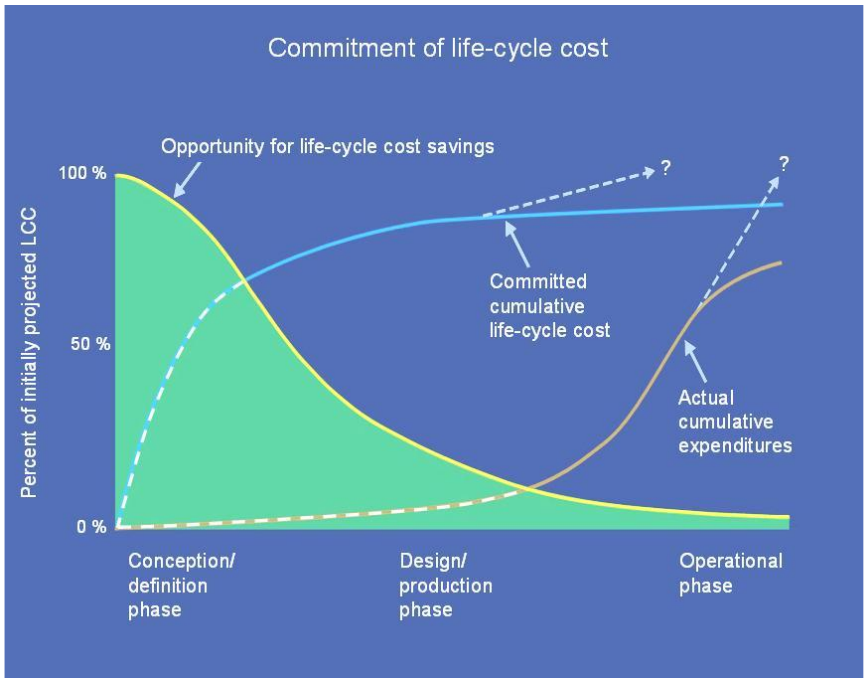
- influencing Product design
- developing the most effective support solution
- defining support resource requirements

This chapter is in direct relation with the iterative work needed for developing an effective support solution, and definition of support resource requirements.

General objectives for the Product must be translated into more specific requirements for each project. The key to a productive LSA and the ability to develop a cost-effective Product support is to concentrate available resources on activities that are mostly beneficial to the project. This is defined and tailored in an analysis strategy. It is necessary to consider the type and scope of the development project when choosing an analysis strategy. Influence on Product design can also vary due to previous design decisions and the life cycle phase.

The opportunity for influencing design in order to fulfill supportability requirements and reduce Life Cycle Costs (LCC) is at its highest in the beginning of a project, during the conceptual phases (refer to Fig 1). Influencing design as early as possible can reduce or eliminate the need for later changes in design (eg, need for redesign) to make the Product fit for operation and support. Supportability requirements are considered useful for designing a new Product with

respect to total system availability and cost effectiveness. The purpose is to influence the design with supportability requirements in a similar way as the primary Product design and its requirements influence the support system design. Reviews and baselines produce these results within a project.



ICN-B6865-S3000L0077-001-01

Fig 1 Opportunity to influence design and support cost during a Product life cycle

Integrated design teams, Product design and requirements benefit both the Product and the customer. LSA must be considered as an integral part of systems engineering.

Product design influences the logistics footprint, the physical size and distribution of support resources within the supporting logistics system. A structured iterative, interactive closed loop is necessary between design disciplines and LSA disciplines.

1.3 Scope

This chapter describes:

- design parameters to be influenced by LSA and vice versa
- how to make a strategy for LSA to influence Product design
- the perspectives of supplier and vendors
- reviews of milestones
- the benefits of LSA influence on Product design

The approach to influence the design with the parameters of supportability is applicable on systems and subsystems within the Product, as well as on the design and development of support systems.

2 Design considerations

Many considerations can influence the design of a system/subsystem or equipment to make the complete Product more efficient and cost effective. Such considerations are:

- Availability. Refer to Para 2.1.
- Reliability. Refer to Para 2.2.

- Maintainability. Refer to Para 2.3.
- Testability. Refer to Para 2.4.
- Prognostics. Refer to Para 2.5.
- Standardization. Refer to Para 2.6.
- Interchangeability. Refer to Para 2.7.
- Environmental considerations. Refer to Para 2.8.
- Human factors/ergonomics. Refer to Para 2.9.
- Obsolescence. Refer to Para 2.10.
- Supportability. Refer to Para 2.11.
- Cost effectiveness. Refer to Para 2.12.
- Software design. Refer to Para 2.13.

2.1 Availability

Availability is the measure of the degree to which an item is in an operable and ready-for-use state at the start of a mission or operation when the mission or operation is called for at an unknown time. This can also be referred to as "operational readiness".

Reliability, maintainability, and supportability characteristics contribute to the availability of the Product. Refer to Fig 2.

The **inherent availability** of a Product depends on its reliability and maintainability characteristics. It is the probability that a system will operate at any point in time when used under stated conditions in an ideal support environment (eg, no lack of support resources). It excludes preventive maintenance, delay times and is expressed as:

$$A_i = \frac{MTBF}{MTBF + MTTR}$$

Where:

- A_i is the inherent availability
- *MTBF* is the mean time between failure
- *MTTR* is the mean time to repair

Achieved availability is similar to inherent availability, with the exception that preventive maintenance is included. It is expressed as:

$$A_a = \frac{MTBM}{MTBM + \overline{M}}$$

Where:

- A_a is the achieved availability
- *MTBM* is the mean time between maintenance
- $\overline{M}$ is the mean time for active maintenance activities

Operational availability is achieved by combining the Product and its support system. It is described as the probability that a Product will operate sufficiently when called for, provided it is used under stated conditions in an actual support environment. It is expressed as:

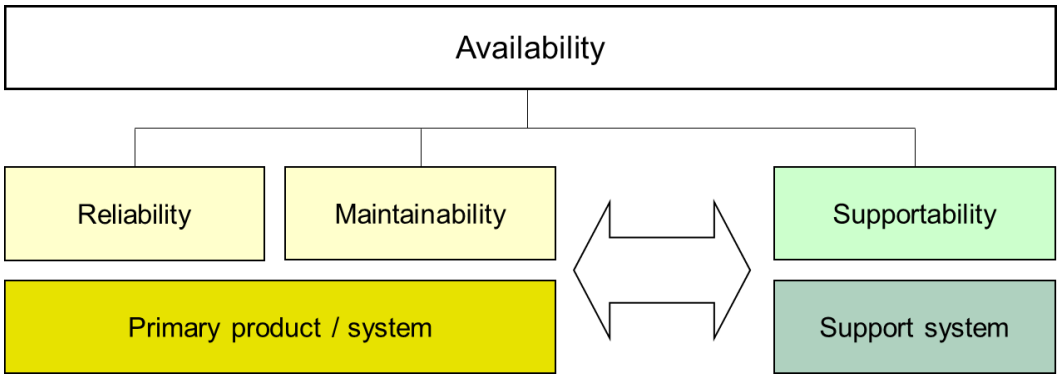
$$A_o = \frac{MTBM}{MTBM + MDT}$$

Where:

- A_o is the operational availability
- *MTBM* is the mean time between maintenance

– *MDT* is the mean maintenance down time [1]

Defining the requirements for both the Product and the support system effectively achieves Product availability by fulfilling these requirements through adequate maintenance activities.



ICN-B6865-S3000L0078-001-01

Fig 2 Breakdown of availability as an approach to structure design influence

2.2 Reliability

Reliability is a prime driver of support resources. It indicates how long and how likely the Product will perform under stated conditions without any failure, or the probability that an item can perform its intended function for a specified period of time under stated conditions.

Products with high reliability are usually cost effective in terms of support. In case it is not possible to obtain a high level of reliability, the design of the Product must compensate this issue by providing easy failure location and easy performance of maintenance activities. To some extent, it is possible to compensate low reliability with increased maintainability and supportability in order to ensure continuing availability.

2.3 Maintainability

Maintainability measures the ability of an item to be held or restored to a specific condition, when qualified personnel perform maintenance, using recommended procedures and resources, at each recommended level of maintenance and repair.

Characteristics of maintainability are, for example:

– the time needed to replace an equipment or part in a system
– the repair of equipment in order to restore or maintain its functionality
– the amount and complexity of support resources that are required
– the required level of competence for personnel performing the activities

Product design can include various degrees of protection against environmental conditions, depending on different degrees of harshness. For example, the design can be specific to withstand an operational and maintenance environment in airports/airbases, loading/unloading with support equipment in sandstorm, hail or salt-laden environments. Refer to Chap 8.

Refer to Chap 13 for maintainability aspects for software, such as software loading.

[1] Availability definitions from Benjamin S Blanchard, *Logistics Engineering and Management,* 6th edition, Prentice-Hall, Inc., Englewood Cliffs, New Jersey 07632

2.4 Testability

Testability is a design characteristic, which identifies and establishes the status of an item (operable, inoperable, or degraded) and the location of any faults/failures within the item in an effective and timely manner.

It is possible to achieve a design solution to fulfill testability requirements by means of a test system integrated into the Product (eg, Built-In Test (BIT)), or by a test system as a part of the support resources requiring an interface to the Product to locate failures. Carefully designed test systems reduce the effort to detect, localize and correct failures/faults, reduce the number of No-Fault-Founds (NFF) and can also be used to verify full system functionality.

Redesign driven by testability requirements is a highly complex activity. It is fundamental to add milestones in the design program to allow for necessary iterations. Testability requirements can also be driven by the results of failure mode analysis. Refer to Chap 7.

2.5 Prognostics

Product parameters and data can be used in models to predict the need for maintenance or repair. The functionality of maintenance or repair prediction can either be part of the Product or the support system.

The benefits of prognostics include a deeper knowledge of Product's "health" status, which can help avoid critical failures and plan for maintenance activities to ensure system functionality.

In systems that do not have a means of prognostics, preventive maintenance is used to prolong the system lifetime and safety, and is balanced against the corrective maintenance activities and resources.

2.6 Standardization

Using standard equipment instead of special equipment often proves to be more cost effective. Development/design cost and time is reduced when using an existing part or system that fulfils the requirements.

Using standardized equipment provides for the use of existing support resources that fulfill the requirement, thus reducing the need for investments in design and acquisition of support resources. It is necessary to carefully consider design choices that require specially developed support equipment, since such equipment increases costs.

Other positive effects include an increased maturity and knowledge of the Product/system and an increased operational unit mobility, by keeping a small logistics footprint.

Factors that support the potential benefits of using standard equipment or Commercial off The Shelf (COTS) include:

- the use of existing equipment avoids the development costs for new support resources
- avoiding costs for the development of new training programs
- using common support resources to increase the availability of support resources and reduce the logistics footprint
- using standardized equipment to reduce the time required to determine and develop support resource requirements
- using support and test equipment to increase personnel proficiency when personnel use the same equipment more often instead of having to learn to use different equipment

Standardization also improves the interchangeability characteristics of the Product.

Software design can also be influenced by standardization requirements concerning, for example, programming languages, information structures or software, and information carrying media.

Before starting the design effort, it is necessary to establish requirements on standardization to achieve the benefits and minimize the cost of designing or redesigning to meet requirements.

2.7 Interchangeability

A Product design that enables interchangeability of equipment, components and parts within or between Products is recommended, where applicable. This provides for increased flexibility of usage of the equipment and a reduction of spare stock levels.

Before starting the design effort, it is necessary to establish requirements on interchangeability to achieve the benefits and minimize the cost of designing or redesigning to meet requirements.

2.8 Environmental considerations

Available information on the operational and maintenance environment can influence the choice of equipment and design solutions. Temperatures, humidity, sandy and salty environments are some of the parameters that influence operation and maintenance.

Ensuring the Product fulfils requirements in-between operation during transportation, assembly/disassembly, and storage is also useful to increase the robustness of the Product.

Chemicals/materials needed for maintenance, operation and disposal are also important from an environmental perspective. Including hazardous materials, hazardous waste and environmental pollutants has an impact on the Life Cycle Cost (LCC), as well as on the safety of personnel in contact with the Product. Refer to Chap 16.

2.9 Human factors/ergonomics

It is necessary to identify Product requirements related to human factors and ergonomics. Product design must consider accessibility for operators and maintainers. Activities performed during operation and maintenance, which include human factors must be considered as well. Handling qualities, for example, parameters such as weight and size are relevant to support resources to be able to perform activities physically.

Design tools can be used for early analysis of the accessibility for human and definition of assembly/disassembly activities in virtual models.

Chemicals/materials included in the Product or needed for maintenance, operation and disposal are also important from the point of view of human factors. In fact, including hazardous materials and environmental pollutants have an impact on the safety of the personnel and the need for support resources.

Product interface requirements, such as access panels for maintenance personnel, and operator stations requirements are a shared interest between LSA and Product design. Refer to Chap 6.

2.10 Obsolescence

Depending on the expected operational life of the Product, it is recommended that the possible obsolescence situation during the design phase be considered. In case obsolescence occurs, specific actions will be necessary to ensure continuous functionality. Typical actions include redesign/modification, purchasing activities during Product lifetime, scrapping of a system and replacement with a new system. When designing or choosing systems or subsystems during the Product design phase, it is useful to keep in mind the future replacement of an item due to obsolescence. Refer to Chap 15.

2.11 Supportability

Supportability is the measure of the degree to which all resources required to operate and maintain the Product are designed properly and available in sufficient quantity. Supportability encompasses all IPS elements, such as technical information, support equipment, spares and personnel.

Careful system design that considers maintainability and reliability helps reduce the amount of resources required for Product support. It is also possible to reduce Product downtime (eg, active maintenance time), logistics delay times, and administrative delay times. A Product that requires many support resources has a higher risk of shortage of support resources, waiting time or queuing for resources. MTA defines the resource needs. Refer to Chap 12.

LORA defines the characteristics of the support solution by evaluating Product maintenance requirements, resource needs and logistics footprint limitations against availability, operation, and support cost targets. Refer to Chap 11.

2.12 Cost effectiveness

The design and relevant choices have a cost implication for the LCC. Cost implications can be broken down into design/development, production/procurement, operational and disposal costs. It is necessary to perform comparative analyses of different design solutions, including the relevant support resources. These characteristics and costs are necessary to balance availability and LCC (refer to Fig 3). Refer to Chap 14.

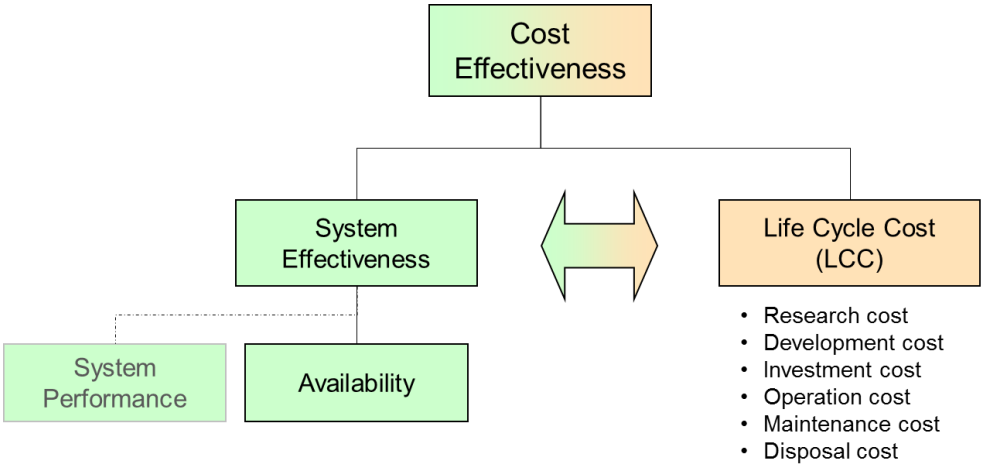


Fig 3 Effectiveness - balancing availability and LCC

2.13 Software design

Software is usually a part of modern Product design. Software design requires careful considerations, in order to include in the software some of the features mentioned in this chapter.

Considering the consequences of software support activities described in Chap 13, decisions on software design and implementation are necessary to fulfil Product availability requirements.

3 Development programs

LSA must be integrated in Product development projects, to ensure that the corresponding LSA activities are acknowledged and performed in a documented, structured, and interactive closed-loop process.

The LSA representative is an important member of the entire project team.

3.1 Strategy for LSA influence on design

The key to a productive and cost-effective analysis effort is the concentration of available resources on activities that most benefit the program.

LSA aims to influence design, develop the most effective support solution, and define requirements for support resources. These general objectives need to become more specific for

individual projects. The type of project determines the amount of design flexibility and opportunity to influence the design, and is the main input when deciding on a strategy.

It is necessary to review, refine and balance defined objectives and the analysis strategy against available resources, until they become definite program goals or requirements.

3.2 Design influence on development programs

The very nature of development programs can vary from:

- new development programs, where the Product and support products are developed from the very beginning
- design changes/improvements for systems or subsystems of an existing Product
- fast-track programs using existing technology developed in-house or by a supplier

Therefore, the amount of design freedom and possible design influence differs depending on the nature of the development program. Development programs for complex Products are often a combination of the development types described above. The program will also prioritize LSA effort and objectives accordingly.

Furthermore, design freedom can be available only for the support system but not for the Product, and vice versa. The LSA objective of making reliability, maintainability, and supportability requirements an integrated part of Product requirements and design can best be achieved if designers are oriented towards reliability, maintainability, and supportability objectives commencing with the design effort.

3.3 Supplier design

Supplier design is design performed by a vendor or subcontractor and must be approached in the same manner as in-house design.

It is important to provide the supplier with LSA goals and requirements specific to the supplier in order to influence supplier design before starting the design efforts.

In conjunction with design influence by goals and requirements, the requiring party must initially decide and specify the LSA activities that must be performed by the supplier, which activities must be shared between the requiring party and the supplier, and those that must be performed solely by requiring party. Once done, the LSA portion of the contracting plan can be developed and work requirements written into the procurement documentation. It is often useful to allow the prospective performing activities, under the bidding terms of the procurement, to recommend adding or deleting LSA activities and to provide a more detailed subtask definition and schedule.

Additionally, prospective performing activities can be encouraged to make use of cost-effective data generation procedures. Acquisition program objectives must be considered in preparing procurement documents. For example, in technology demonstration procurement, certain LSA activities can be specifically excluded. Supportability objectives for this type of procurement would best be served through design influence and generation of LSA data for subsequent detailed analysis efforts when the technology is utilized. If the acquisition program is oriented to develop and procure a Product, then other LSA activities become equally important.

When a supplier provides a COTS system or equipment, the possibility to influence the equipment design is obviously very limited. However, the influence on the integration of the system/equipment in the Product and the support system is still as important as for any other development project.

3.4 Critical/milestone design reviews

It is essential to establish and document design review procedures in case procedures do not exist already. They will provide official review and control of released design information jointly with the LSA program, in a timely and controlled manner.

These procedures define accept/reject criteria, the method of documenting reviews, the types of design documentation subject to review, and the degree of authority of each reviewing activity.

Program planning must coordinate and integrate the design and development reviews and the LSA reviews. Formal review and assessment of LSA requirements are an integral part of each Product design review. It is necessary to document results of each Product design review. Design reviews must identify and discuss all pertinent aspects of the LSA program.

Technical information generated and documented during the design process must be shared with designers and supportability specialists, in order to identify interface problems between design concepts and operators, maintainers, and support resources. Design documentation must include technical design information such as diagnostic features, interfaces, reliability estimates, obsolescence evaluations, and item functions determining supportability.

It is recommended that subcontractors and suppliers, as appropriate, be included in all scheduled "bottom-up" reviews. Developing and coordinating agendas helps address the relevant topics, as they apply to the program phase activities and the reviews.

Examples of topics are:

- LSA performed using task and WBS element
- LSA assessment of proposed design features including supportability, cost, and readiness drivers, and new or critical support resource requirements
- corrective actions that have been considered, proposed, or taken, such as:
  - support alternatives under consideration
  - system/equipment alternatives under consideration
  - evaluation and trade-off analysis results
  - comparative analysis against existing products
  - design or redesign actions proposed or taken
- review of LSA requirements - supportability (with review of specifications as developed)
- progress toward establishing or achieved goals
- required, completed, and scheduled LSA documentation
- design, schedule, or analysis problems affecting LSA

Incorporating design reviews during the Product's in-service, are also encouraged to improve the Product based on operational experience and to effectively sustain a competitive Product. In-service LSA deals with the principles and means for such development. Refer to Chap 17.

4 Checklists

To facilitate design work and reviews, checklists can be used as a starting point and input. However, these checklists serve as examples. It is vital to define the questions and adjustments to the checklist to suit the program. A synergy between LSA and design disciplines is often necessary to define those checklists.[2]

4.1 Selection of parts/equipment

A checklist for the selection of parts/equipment can include, but is not limited to:

- Have appropriate standards been consulted for the selection of parts?
- Have the selected parts/equipment been evaluated in terms of reliability, maintainability and supportability?
- Have suppliers been selected for component part procurement?
- Is the supplier reliable in terms of quality, timeliness, and cost effectiveness?

[2] Checklist developed from Benjamin S Blanchard, *Logistics Engineering and Management,* 6th edition, Prentice-Hall, Inc., Englewood Cliffs, New Jersey 07632

4.2 Reliability

A checklist for reliability can include, but is not limited to:

- Has the system/equipment wear-out period been defined?
- Have failure modes and effects been identified?
- Are item failure rates known?
- Have parts with excessive failure rates been identified?
- Has mean life been determined for the system/equipment?
- Has equipment design complexity been minimized?
- Is protection against secondary failures (resulting from primary failures) incorporated where possible?
- Are reliability requirements met?

4.3 Maintainability

A checklist for maintainability can include, but is not limited to:

- Has the number of different kinds of fasteners been minimized?
- Is the type of fasteners standard items?
- Have the fasteners been selected based on the requirements for standard tools rather than special tools?
- Are equipment items identified as replaceable?
- Has replacement time been minimized?
- Has the operational environment (eg, hailstorms, sandy, salty environments) been taken into consideration?
- Have considerations been made on how to load software, loading times and media?

4.4 Testability

A checklist for testability can include, but is not limited to:

- Have self-test provisions been incorporated where appropriate?
- Is the extent of self-test compatible with LORA?
- Is the self-test automatic?
- Have direct fault indicators been provided? (eg, warning lights, messages)
- Are test points/interface provided to enable checkout and fault isolation beyond self-test level?
- Are test points/interface accessible?
- Are test points/interface functionally or conveniently grouped to facilitate sequential testing?
- Are test points/interface provided for direct test of replaceable items?
- Are test points labeled?
- Can every equipment malfunction be detected by a go/no-go indication at system level?
- Will software provide adequate test information?

4.5 Prognostics

A checklist for prognostics can include, but is not limited to:

- Is it possible to identify the functionality to predict maintenance needs for the equipment?
- Is it possible to identify and obtain the parameters used to predict maintenance needs?
- Do parameters used for prognostics have an adequate sampling frequency?

4.6 Standardization

A checklist for standardization can include, but is not limited to:

- Are standard equipment/parts incorporated in the design as appropriate?
- Are the same parts used in similar applications?
- Has the number of different part types used throughout the design been minimized?

- Are identifying labels, markings and nomenclature standardized in full measure?

4.7 Interchangeability

A checklist for interchangeability can include, but is not limited to:

- Are modules and components with similar functions interchangeable electrically, functionally and physically (ie, the same form, fit and function)?
- Are components with the same part number interchangeable, even if different suppliers provide them?

4.8 Environment

A checklist for environmental aspects can include, but is not limited to:

- Are the hazardous materials and pollutants identified and minimized?
- Is it costly to handle, store, disassemble or dispose of the materials and equipment chosen for the design?
- Are special containers or facilities necessary to handle hazardous materials or pollutants?
- Are the selected materials and equipment consistent with environmental regulations on operation and disposal?

4.9 Human factors/ergonomics/accessibility

A checklist for human factors, ergonomics and accessibility can include, but is not limited to:

- Are doors provided where appropriate? Are they hinged?
- Are the size and location of openings adequate for access?
- Are the doors and openings labeled? What information does the label provide?
- Are access door fasteners minimized?
- Are quick-release fasteners used for access doors?
- Are tools necessary to gain access?
- If tools are needed to gain access, are they minimized and standard design?
- Is access between modules and components adequate?
- Are hazardous materials and pollutants identified and minimized?
- Is it necessary to use protective equipment when performing the maintenance task?
- Is it possible to perform maintenance tasks with protective equipment if necessary (eg, gloves, helmet)?
- Is the number of lifting devices for heavy or bulky items minimized?
- Compared with the working position, is the time it takes to carry out a maintenance task reasonable?
- Are access requirements compatible with the frequency of maintenance?

4.10 Obsolescence

A checklist for obsolescence can include, but is not limited to:

- Is the item at risk for obsolescence during the lifetime of the Product?
- Does the design facilitate redesign if necessary due to obsolescence?
- Is it possible to emulate or recreate the design?
- Are any aftermarket sources available?
- Is the supplier required to inform and initiate a purchase process or redesign of parts to cover the rest of the Product lifetime?

5 Associated parts of the S3000L data model

The documentation of the associated data to this chapter is supported by the following Units of Functionality (UoF), refer to Chap 19:

- S3000L UoF Design Change Request

Chapter 6

Human factors

Table of contents

List of tables

References

Table 1 References

Chap No./Document No.	Title
S6000T	International specification for training analysis and design
MIL-STD 1472F	Human engineering, design criteria for military systems, equipment and facilities
1272/2008/EC	EU Regulation on classification, labelling and packaging of substances and mixtures (CLP)
2011/65/EC	Directive on the restriction of the use of certain hazardous substances in electrical and electronic equipment (RoHS2)
1907/2006/EC	Regulation concerning the Registration, Evaluation, Authorization and Restriction of Chemicals (REACH)
2006/42/EC	Directive on machinery

Chap No./Document No.	Title
2009/104/EC	Directive concerning the minimum safety and health requirements for the use of work equipment by workers at work
ISBN 9780415304306	International Encyclopedia of Ergonomics and Human Factors
ISO 26800	Ergonomics - General approach, principles and concepts
ISO 6385	Ergonomics principles in the design of work systems

1 General

1.1 Introduction

Human factors provide source data that must be used within LSA activities to determine maintenance crew and support equipment requirements. This relationship begins in the design process and continues through the entire life cycle, with the focal point during Maintenance Task Analysis (MTA).

1.2 Objective

It is necessary to coordinate LSA, maintainability and supportability functions to ensure that potential support solutions are within established support thresholds (including the requirements of human factors). This is accomplished by considering the crew size and the required support resources for maintenance and operational support tasks. Human factors limitations influence the establishment of support environment as well as the design of the Product. The limited capabilities of a human being determine the limitations on Product usage and support. With respect to support in particular, human factors have significant influence on the practicability of operational support and maintenance.

1.3 Scope

The capabilities and natural limitations of a human being should influence design and LSA. This chapter aims to support design and supportability engineers who perform technical and supportability analyses and describes the relationship and integration between human factors and the LSA process. Because human factors can influence various phases of the LSA process it is recommended that the results of the human factors analysis, including warnings and cautions, and the need for specific equipment to mitigate hazardous tasks/conditions be documented and available at a very early project phase.

2 Logistics support analysis and human factors

2.1 Physical abilities and limitations

The LSA and human factors integration occurs throughout the whole Product life cycle. Product modifications or proposed design changes can lead to changes to the maintenance activities that include human factors constraints and/or limitations.

LSA and human factors have a common objective, but specific purposes and goals. Human factors focus on various aspects, for which several standards (refer to Table 1) provide rules and guidelines, including:

- anthropometric aspects
- ergonomic aspects
- other physiological aspects
- psychological aspects
- legal requirements

The diameter of the average human forearm, for example, is a human factor. Any access panel that requires a reach beyond the wrist must conform to the minimum size indicated in the relevant standards. In this case, the human factors influence the design to ensure all these access panels comply with this size requirement. LSA must take into account these standards to evaluate potential support solutions. In addition, LSA must consider the need for special support equipment to complete a task under limited moving space conditions. If alternate design solutions are identified during the analysis activities, they can be presented back to design for reconsideration.

2.2 Limitations due to hazardous conditions

Another aspect concerning human factors is the limitation of human activity because of health threat.

The handling of dangerous or hazardous materials must follow strict regulations to ensure the safety of all persons involved and the environment. At the end of the Product life cycle and during the disposal phase in particular, human factors can be crucial due to an increased need to handle material, which is a risk for human health.

Other limitations must be considered for work under extreme environmental conditions such as:

- cold or hot environments
- humid environment
- working underground or underwater
- critical environment due other reasons (eg, dust, exposure to fumes, sound, light)

Design guidelines must ensure the protection of human beings against the effects of these environmental impacts. LSA must address these guidelines.

3 Human factors analysis aspects

3.1 Influence on design

There are many industry standards (refer to Table 1) that address human factors design constraints and requirements. The contractor/customer agreements and the contract include and makes reference to them as appropriate.

It is necessary to take into account the human factors qualitative data to determine the maintenance crew skills, size and support equipment needs. For example, human factors will provide weight limits for a single person lift and help identifying the maintenance crew size for a specific task. This analysis will also identify the need for mechanical lifts, maintenance stands or provide a feedback to the designers on the need for grips. This information can be used in a trade-off study for design alternatives as an iterative process. Human factors provide specifications and standards that must be applied to the individual maintenance tasks.

If applicable, the crew size requirement should take into account the demographic aspects as well. This means that an all-male crew can have a greater lift limit than a mixed male/female crew. Therefore, the weight of the item determines the proper number of people required for lifting, but the number will change based on the specific crew demographics. The weight restriction must be taken into account for mechanical lift requirements and therefore impact support equipment requirements. The location can influence the need for maintenance stands, and the type of stand required will depend on the maintenance crew size.

3.2 Guidance for LSA

The LSA Guidance Conference (LSA GC) must identify the rules and guidelines for LSA and human factors analysis, including maintenance crew demographics, lift, reach and other standards and limitations. The customer and contractor must agree on specific standards. In addition to this agreement, it is recommended to review all applicable standards and relevant legislation, and to document any exception.

Note

Exceptions for military equipment are typically only valid during wartime. An ever-increasing number of civil contractors in military support implies that the Product and the support equipment must meet civil regulations, regardless of their use.

LSA receives input from design and development trade-off analysis of each design alternative. From the first design effort, LSA will analyze the design alternatives and compare the relevant support requirements, and make recommendations based on the supportability and Total Cost of Ownership (TCO). Several handbooks such as the “International Encyclopedia of Ergonomics and Human Factors”, ISO standards as well as military specific standards (eg, MIL-STD 1472) provide valuable information (refer to Table 1). It is recommended that the applicability of any local specific rules be verified. The process involves identifying all the possible maintenance actions and takes into account the mental workload under the expected operational conditions. Tired people under a high level of stress can have a shorter memory span. Maintenance sequences must follow a similar logic, and color coding must be consistent. Each maintenance task will result in a list of resources required to accomplish the task. These resources include spare parts and consumables, maintenance man-hours, training, support equipment and test equipment. The specific considerations on human factors include the number of maintainers and the amount of support equipment related to human factors.

The entire analysis process is an iterative process that must be used for each design change. Within the LSA process, the more detailed MTA starts upon the Product design approval. Each maintenance task is compared against the human factors requirements to ensure that all maintenance and operational support activities are within the human factor's limitations.

4 Human factors to consider

The human factors are not limited to those listed here, which provide an outline of the aspects that can influence Product design and the design of the support environment (operational and maintenance related), especially in LSA.

4.1 Anthropometric aspects

Anthropometric aspects that influence design include, but are not limited to:

- lines of sight (visual field, vertical and horizontal)
- audio signals requirements
- muscle strength of arms, hands and thumb
- required muscle strength for vertical pull extensions
- required muscle strength for horizontal push and pull movements
- maximum weight of units to be lifted
- maximum weight of support equipment
- arm and hand access dimensions

4.2 Ergonomic aspects

Ergonomic aspects that influence design include, but are not limited to:

- design of controls (eg, switches, cranks, joysticks, ball controls, hand wheels, levers, pedals, knobs)
- minimum handle dimensions
- workspace design (eg, seated, standing, mobile)
- difficult accessibility - ramps and ladders
- doors and access panel dimensions
- illumination requirements

4.3 Environmental aspects

Environmental aspects that influence design include, but are not limited to:

- effective temperature
- limits of extreme cold and warm temperature conditions
- influence of wind-chill on human beings
- decreased performance of human beings under extreme climatic conditions
- ventilation requirements
- exposure limits ultraviolet radiant energy
- exposure limits to pollution like dust, fumes
- noise limitations
- shock current intensities

5 Associated parts of the S3000L data model

The human factors analysis activities are performed outside the LSA scope. The results are a precondition to enable the determination of a specific working environment for the Product operators and maintainers.

Human factors analysis data is not documented in the frame of S3000L data model and there are no related Units of Functionality from Chap 19.

Note

Human factors analysis outputs also serve as inputs to human factors analysis in S6000T.

6 Additional information

Each project must determine human factors requirements for the analysis process. Table 2 provides sources for more detailed information on how to evaluate the support equipment related to human factors for any project.

Table 2 Sources for additional information

Source	URL
Human Factors and Ergonomics Society (HFES)	http://www.hfes-europe.org
Designing for humans	http://www.designingforhumans.com

Chapter 7

Corrective maintenance analysis

Table of contents

List of tables

List of figures

References

Table 1 References

Chap No./Document No.	Title
Chap 3	LSA process
Chap 4	Product structures and change management in LSA
Chap 8	Damage and special event analysis
Chap 11	Level of repair analysis
Chap 19	Data model
S4000P	International specification for developing and continuously improving preventive maintenance

1 General

The process for corrective maintenance analysis described in this chapter is a part of the event-driven analysis activities, which are performed as part of the LSA. Corrective maintenance analysis aims to identify the corrective maintenance and failure mode isolation tasks, which are necessary to develop and document.

This chapter describes possible methods to evaluate and identify corrective maintenance task requirements that must be performed on the Product and/or its equipment in case an inherent failure occurs.

Corrective maintenance analysis identifies:

- corrective maintenance tasks required to repair and restore the Product and its equipment after a failure mode has occurred
- symptoms indicating that a failure mode has occurred
- failure mode isolation tasks required to determine which specific failure mode has occurred, in case where two or more failure modes have the same symptoms but require different corrective actions

Note

Corrective maintenance analysis does not include step-by-step descriptions on failure mode localization procedures. It only identifies the tasks needed to determine that a specific failure mode has occurred, or the tasks that can narrow down an ambiguous

symptom to a smaller set of possible failure modes, in case it is not possible to isolate a specific failure mode.

Failure Mode and Effect Analysis (FMEA) or Failure Mode Effect and Criticality Analysis (FMECA) are the starting point for the corrective maintenance analysis. In many cases, FMEA/FMECA is performed during the Product Design & Development (D&D) phase, and there are two possible approaches:

- The equipment FMEA is a bottom-up analysis methodology that starts with equipment failure modes, their local effects, and their higher effects on the equipment under analysis
- The system/functional FMEA is a top-down analysis methodology that starts from the functions of the product/system under analysis, its functional failure modes, the functional failure effects and the respective failure causes identified for each functional failure mode

Note

This chapter does not provide any guidance on how to carry out an FMEA/FMECA, but it defines this type of analysis as a prerequisite. If there is no FMEA/FMECA in place, it is necessary to determine how to develop one, or to define an alternative method on how to progress with the corrective maintenance analysis.

1.1 Purpose

Corrective maintenance analysis aims to describe a method to evaluate and identify corrective maintenance task requirements during the early phases of Product D&D. This method also evaluates the Product and its equipment from a testability perspective in order to identify additional needs for failure mode detection and/or isolation means, and/or design improvements.

Depending on the overall support concept, corrective maintenance can include tasks to be performed on the Product itself, as well as tasks to be performed on the equipment when removed from the Product.

Corrective maintenance at Product level can be limited to the replacement of the faulty equipment in order to reduce down time and limit the required resources (eg, special support equipment and/or specific personnel competences). After that, it is possible to either repair the replaced equipment at the workshop (at any maintenance level), or discard it. If required, it is necessary to define and elaborate on-Product repair tasks for non-replaceable equipment (eg, items of Product structure).

It is not possible to schedule corrective maintenance, which in most cases must be performed as soon as non-acceptable tolerances are reached or exceeded.

Note

For the evaluation and identification of corrective maintenance driven by induced failures caused by special events and/or by damages within or outside the specified usage scenario, refer to Chap 8.

All Product systems, subsystems, equipment and components liable to fail can become candidates for corrective maintenance analysis.

There are different ways to detect a failure mode, for example:

- during Product operation by the operating crew
- during the performance of preventive maintenance
- by an integrated test-/monitoring system
- by chance (random)

Some failure modes can be detected but not unambiguously localized. Therefore, failure mode isolation tasks can be required to uniquely determine that a specific failure mode occurred, in order to replace and/or repair the correct item.

The need for failure mode isolation tasks is primarily dependent upon the availability of built-in detection means and measurement points, but also on the availability of failure mode effect descriptions identified during FMEA/FMECA. These factors come into play when analyzing and determining whether each individual failure mode has a unique set of symptoms or it requires an additional failure mode isolation task.

A failure mode isolation task is a separate task that can support a failure mode localization process. Typical failure mode isolation tasks include the use of external test equipment, functional tests and visual inspections.

1.2 Scope

The scope for corrective maintenance analysis is to describe a method that covers:

- The identification of corrective maintenance and failure mode isolation task requirements to perform on the equipment
- The identification of corrective maintenance and failure mode isolation task requirements to perform on the Product

Note

During the in-service phase, the maintainer will detect a symptom that triggers a fault isolation/localization process to identify the faulty element. The corrective maintenance analysis starts with the analysis of the individual failure modes, and aims to define how the maintainer will detect, isolate and finally restore the faulty element. Corrective maintenance analysis defines all data needed for this process.

Note

The detection and isolation of the failure modes mainly concern the equipment and its installation on the Product. However, the identification of failure modes on structural parts mostly occurs through inspections or Non-Destructive Testing (NDT), regardless of them being preventive or by opportunity.

This chapter mainly focuses on the equipment and its components whether installed on the Product or in the workshop.

1.3 Terms, abbreviations and acronyms

Table 2 provides a list of terms specific for this chapter.

Table 2 Terms specific for this chapter

Terms	Definition/example
component	A non-repairable part of an equipment or sub-equipment.
equipment	Equipment is a part that is fulfilling a specified function or purpose related to the Product.
equipment installation details	Parts necessary to equipment installation on the Product. Example: Fasteners, connectors
failure	Failure is the functional performance deficiency manifestation of a fault.
failure cause	Failure cause defines the physical or chemical process that is the reason for the failure mode.
failure mode	Failure mode defines the functional consequence of a fault. Example: low, erratic, or no output from an electrical circuit
Failure Mode Group (FMG)	FMG represents a set of failure modes that leads to the same set of actions to detect, isolate and rectify those failure modes.

Terms	Definition/example
fault	Fault defines the unacceptable state of the failed item. **Example**: a broken wire
fault isolation process	Fault isolation process is a process that identifies the item at fault.
fault isolation task	A fault isolation task is a task that can contribute to determine the item at fault.
general installation details	Parts of the Product that connect pieces of equipment for a specific function (eg, piping, wiring)
inherent failure	Inherent failure is a failure due to the existing and inherent characteristics of an item.
installed equipment	Equipment located in a specific place within the Product
observable symptom	Observable symptom is a symptom that an operator can recognize as the manifestation of a fault condition. **Note** Flight crews, maintenance personnel or mission crews can recognize an observable symptom.
sub-equipment	A repairable/restorable part of the equipment.
symptom	Symptom is a measurable or visible parameter that, if present, can be associated to a fault, either directly or indirectly.
symptom ambiguity group	Symptom ambiguity group is a set of items that require further fault isolation activities, since the given symptom, or set of symptoms is not enough to determine the actual item at fault.
symptoms signature	Symptoms signature identifies a set of symptoms that, if present, can be associated with a fault.

2 Equipment corrective maintenance analysis

2.1 Introduction

The equipment corrective maintenance analysis aims to define equipment corrective maintenance task requirements, and to identify a method to detect and isolate equipment failure modes.

The result is a set of equipment FMG, which is a unique combination of:

- One corrective maintenance task requirement, which is common to the included (grouped) equipment failure modes (ie, all the included equipment failure modes lead to the exact same corrective action). Some equipment failure modes do not have a viable corrective maintenance task requirement. In such cases, there is just a discard/scrap decision, and possibly an associated equipment neutralization/disposal task requirement. It is necessary to also define this type of conclusion as an equipment FMG.
- One or more symptoms that are common to all equipment failure modes which are grouped against the equipment FMG
- One or more failure mode isolation task requirements that can determine which of the included equipment failure modes has occurred

Note

Any equipment FMG can have more than one defined failure mode isolation task. This is required whenever it is not possible to determine directly and unambiguously which of the failure modes associated with the equipment FMG has occurred. In this case, it is necessary to proceed step by step to reduce the number of possible equipment FMG, in order to eventually isolate only one equipment FMG.

Corrective maintenance analysis starts with the equipment failure modes identified for individual equipment. Each equipment failure mode is then analyzed from the perspective of what corrective maintenance task is required to repair/restore the equipment and whether it is technically and/or economically feasible to perform the corrective task.

Note

In many cases, the Original Equipment Manufacturer (OEM) or equipment supplier performs this type of analysis. This means that the corrective tasks defined by the OEM or supplier only need to be analyzed in the context of corrective maintenance analysis defined for the Product (refer to Para 3).

Note

The result of the corrective maintenance analysis is an important input to the LORA. Refer to Chap 11.

It is also necessary to analyze each equipment failure mode on the basis of the identified symptoms, which can support the detection/isolation of individual equipment failure modes. Symptoms can be built-in test capabilities, measurement points and/or local or next higher failure mode effects as recorded in the equipment FMEA.

If the symptoms defined for an equipment FMG are ambiguous (ie, there is more than one equipment FMG with the same symptoms), then it is necessary to associate the equipment FMG to a failure mode isolation task requirement.

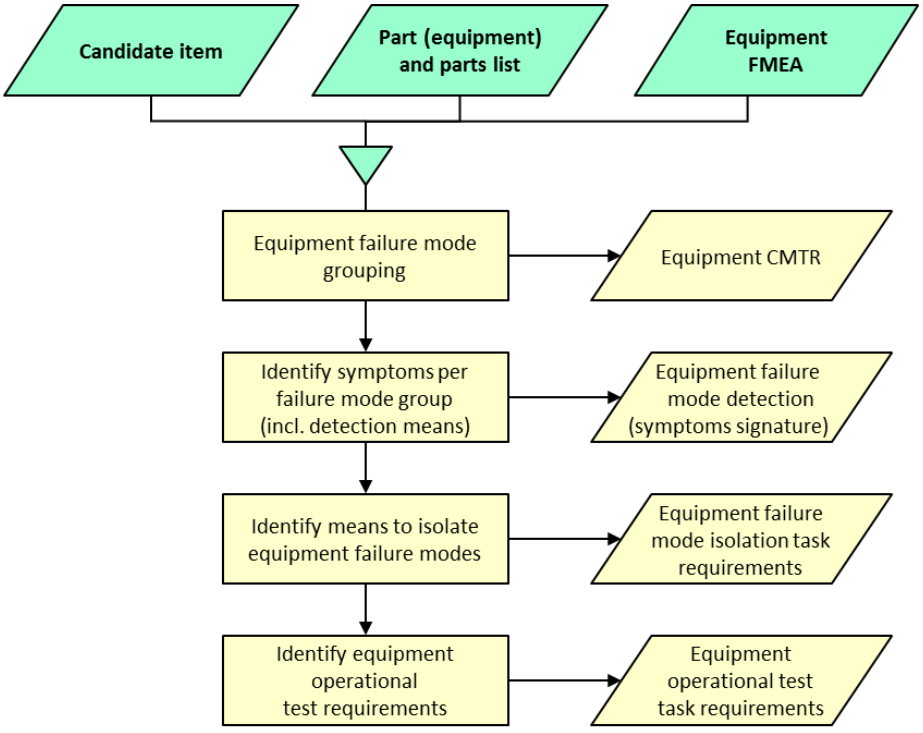
Note

Equipment corrective maintenance analysis does not consider the use of the equipment in any Product, but it only considers the equipment itself.

It is also important to determine the extent of the performance of the equipment corrective maintenance analysis. A general rule is that it is only necessary to define corrective maintenance, failure mode isolation and/or operational test task requirements to be performed on the equipment at customer/operator sites. The analysis often excludes the corrective maintenance performed by industry.

2.2 Method

Fig 1 presents the method for the equipment corrective maintenance analysis in the form of a flow chart.



ICN-B6865-S3000L0080-002-01

Fig 1 Logical flowchart for equipment corrective maintenance analysis

The method for performing equipment corrective maintenance analysis includes the following sub-processes:

1 Perform equipment failure mode grouping and define the corrective maintenance task requirements (refer to Para 2.4).

2 Identify the means (symptoms) for equipment failure mode detection (refer to Para 2.5).

3 Identify the means for equipment failure mode isolation and define the equipment failure mode isolation task requirements (refer to Para 2.6).

4 Identify the need for equipment operational test and define the equipment operational test task requirements (refer to Para 2.7).

Equipment corrective maintenance analysis is an iterative process. If there is no obvious way to group equipment failure modes into one or more equipment FMG based on the resulting corrective maintenance task requirements, at first it is necessary to consider each equipment failure mode as an equipment FMG. Afterwards, it is necessary to compare each equipment FMG to other equipment FMG, to determine the relevant equipment FMG.

2.3 Candidate items

The first activity is to identify equipment which is subject for equipment corrective maintenance analysis. This activity is part of the overall candidate selection activity described in Chap 3. It is necessary to perform it in conjunction with LORA (refer to Chap 11).

Below are some examples of specific questions for the selection of the candidate items for the equipment corrective maintenance analysis:

- Is it possible to open/disassemble the equipment to get access to equipment components that can cause the equipment failure mode?
- Is it possible to replace equipment components with a reasonable effort?
- Is opening/disassembling the equipment covered by the warranty?

- Is it possible to procure the relevant equipment components as spare parts?
- Is it economically meaningful to repair the equipment?
- Does the equipment contain hazardous material, which requires special resources?

2.4 Perform equipment failure mode grouping

Equipment failure mode grouping is a bottom-up approach. This means that if an equipment is an assembly of other repairable/restorable parts, it is necessary to perform the analysis starting from the lowest level of repairable/restorable parts, moving upwards towards the “end” equipment.

Note

The term equipment defines a part that fulfills a specified function or purpose related to the Product.

Note

The term sub-equipment is used when referring to repairable/restorable parts of an equipment, in order to use the phrase "equipment corrective maintenance analysis" for those parts as well.

The rationale for this bottom-up approach is to proceed gradually to determine the FMG and its associated corrective maintenance task requirements for each equipment/sub-equipment. Moreover, the approach ensures that the FMG for the sub-equipment is analyzed when put into the context of the parent equipment.

The differences in the corrective maintenance strategies for the equipment (sub-equipment) will affect the failure mode grouping. Examples of the corrective maintenance strategies are:

- only the test tasks are performed at the customer/operator site, while all other tasks are performed at industry level
- the tasks for test and replacement of the sub-equipment are performed at the customer/operator site, but all other repair and disposal/neutralization tasks are performed at industry level
- all test, repair/restore and disposal/neutralization tasks are performed at the customer/operator site
- all test, repair/restore and/or disposal/neutralization tasks are performed at industry level

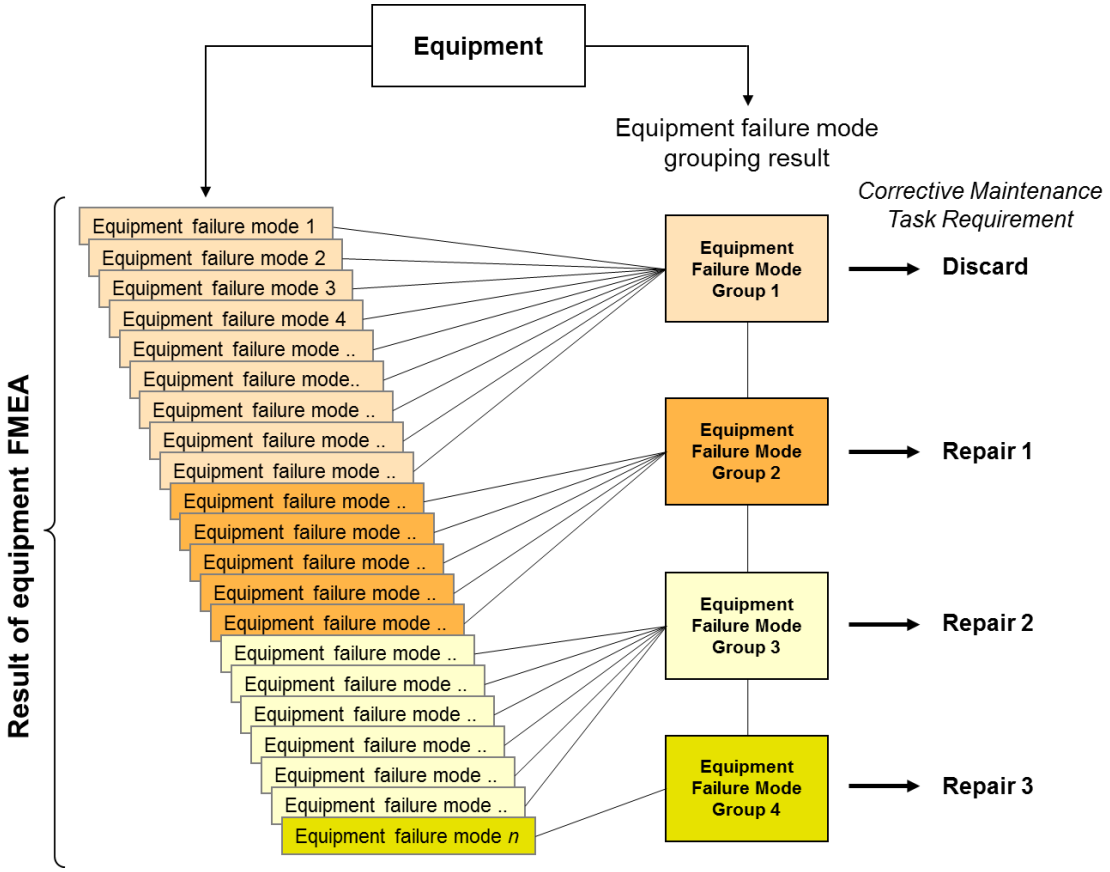
Note

If the approved equipment maintenance concept requires to perform test, repair, restore, disposal and neutralization activities at industry level, then the program must define the need to perform any further corrective maintenance analysis for that equipment/sub-equipment. It is necessary to keep in mind that this analysis provides an opportunity to produce inputs in order to consolidate the equipment maintenance concept from a LORA perspective (refer to Chap 11). It can still be important to keep information on the reparability and testability of the equipment/sub-equipment if the in-service feedback reveals the need for further substantiation or evolution.

Equipment (sub-equipment) can have multiple failure modes that lead to the same FMG. In this case, the number of equipment FMG will be significantly less than the number of failure modes.

In general, the process reuses the existing equipment FMEA/FMECA and equipment part lists (often referred to as EBOM), refer to Chap 4). This information originates from Product D&D activities or from the equipment OEM or supplier.

This process then uses the identified failure modes and parts lists to determine the possibility to develop a corrective maintenance task. If it is possible to repair/restore the equipment (sub-equipment), then its respective failure mode will be associated with an equipment FMG. Then, the latter is associated with a corrective maintenance task requirement that meets the need for the identified corrective action (refer to Fig 2).



ICN-B6865-S3000L0081-003-01

Fig 2 Equipment failure mode grouping

For equipment assemblies, it is necessary to analyze the respective FMG for the sub-equipment in the context of the parent equipment assembly. In this case, it is necessary to consider the respective FMG for sub-equipment in the failure mode grouping activity for the parent equipment assembly, and not the individual sub-equipment failure modes.

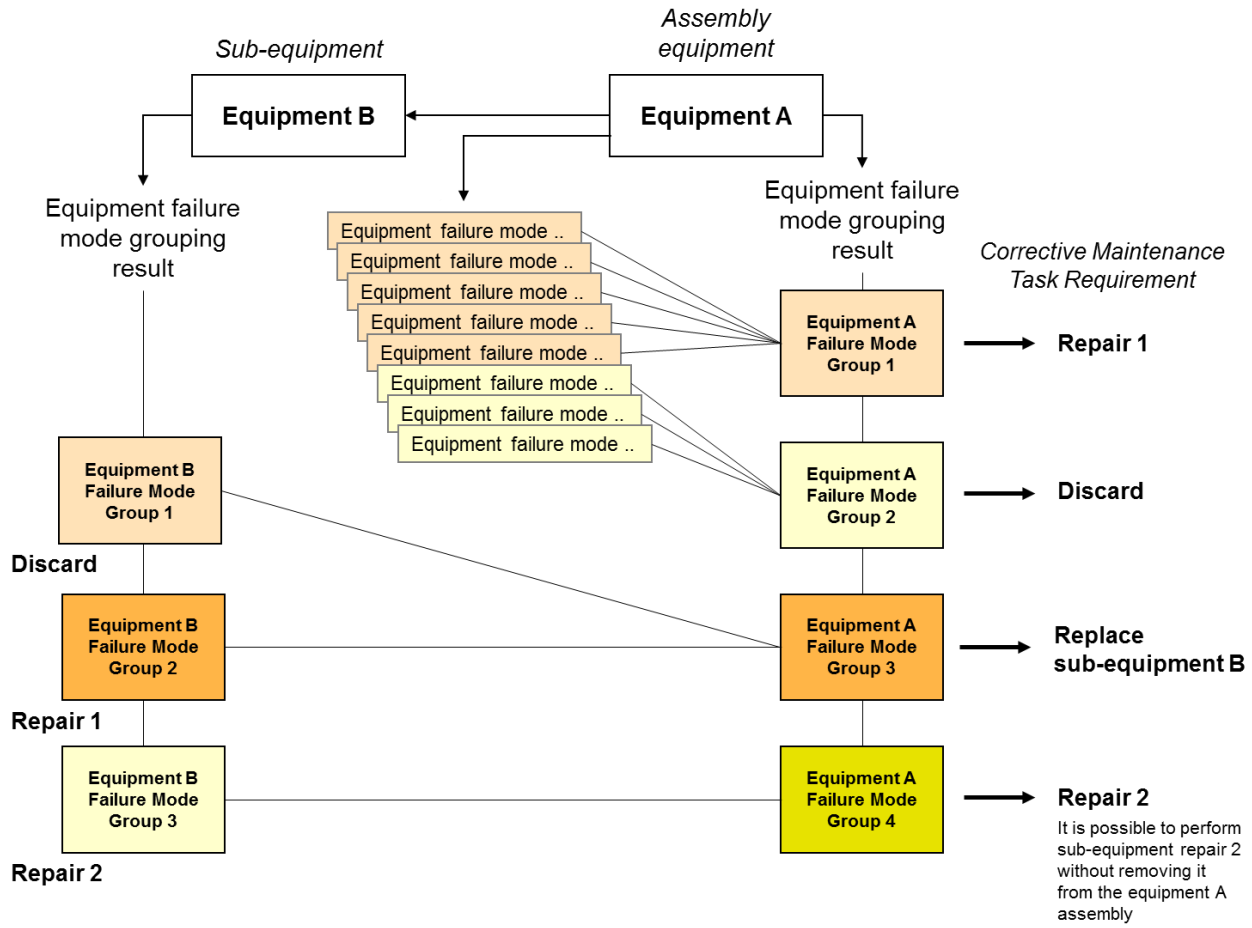
For equipment that contains other equipment (sub-equipment), the failure mode grouping activity must consider:

- Equipment failure modes related to non-repairable/non-restorable components included in the assembly equipment
- FMG defined for included sub-equipment

Note

The FMG defined for the parent equipment must not be the same as the FMG defined for the included sub-equipment. This is because corrective actions for the sub-equipment require a redefinition in the context of the parent equipment assembly, and therefore will be different.

An FMG defined for the assembly equipment can trigger either a replacement of the sub-equipment or a repair of the sub-equipment at equipment assembly level. Since it is necessary to describe the corrective task for the sub-equipment in the context of the equipment assembly, the sub-equipment repair at parent equipment assembly level will be different. For example, the repair of sub-equipment at equipment assembly level can require additional steps to get access to the sub-equipment, as well as different support equipment to gain access.



ICN-B6865-S3000L0148-001-01

Fig 3 Failure mode grouping at equipment assembly level

Note

The example in Fig 3 shows that a computer can be considered as an equipment assembly that contains, for example, a mainboard as one of the sub-equipment. The mainboard itself has its own equipment failure modes and failure mode grouping, which is necessary to consider in the parent equipment (ie, the computer). In case of a failure of a component on the main board (eg, the processor), it is possible to repair the computer by replacing the faulty processor on the mainboard. For this repair, there is no need to remove the mainboard from the computer.

2.5 Identify means or symptoms for equipment failure mode detection

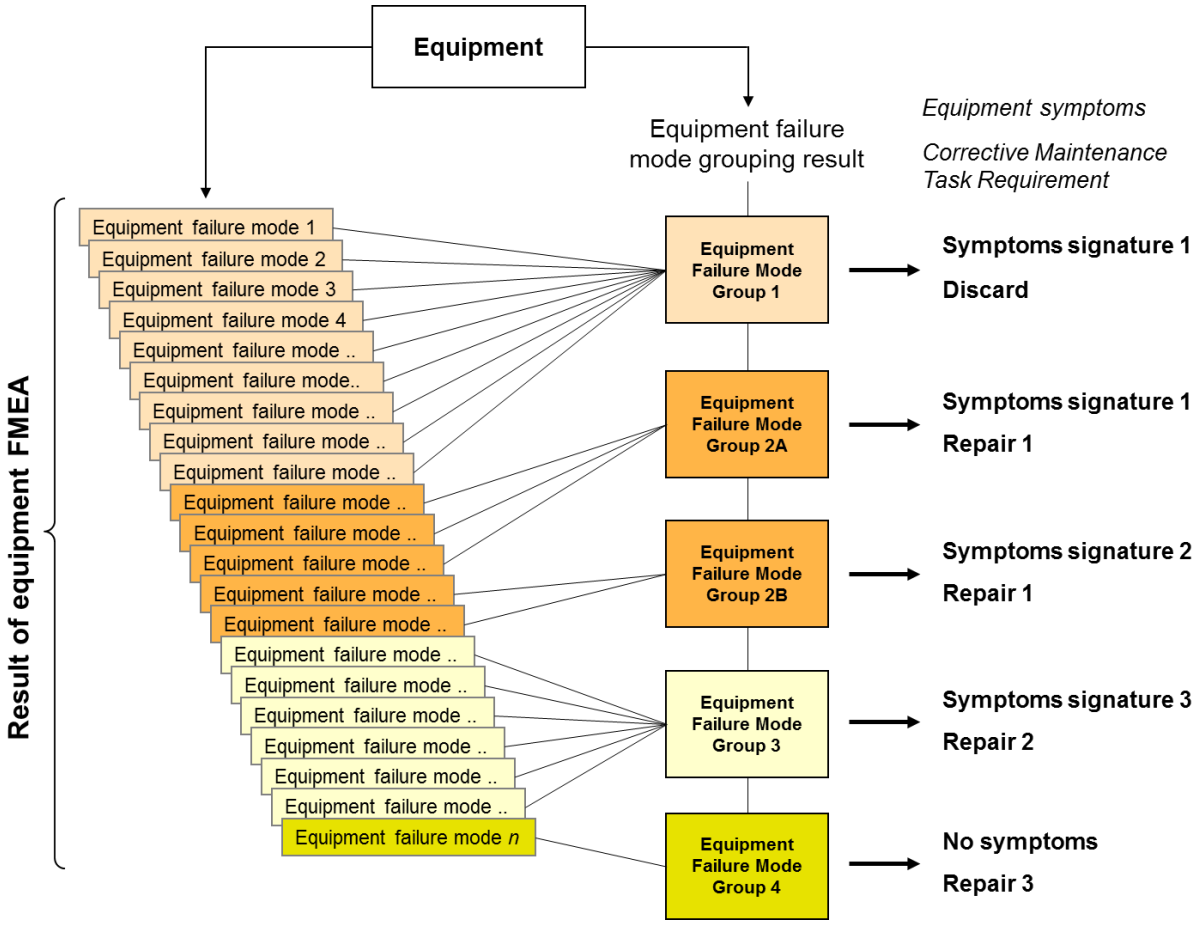
The next step in the equipment corrective maintenance analysis is to identify the means to detect the equipment failure modes. It is necessary to analyze each possible equipment failure mode taking into consideration the following:

- Check if there is any form of automatic detection that triggers a warning device in case a failure mode occurs. If yes, assess the detection rate and false alarm rate of this automatic detection mean.
- Check if there are any measurement points (eg, gauges) that can indicate that the failure mode has occurred
- Check functional symptoms and/or physical symptoms that can help detect the failure mode. It is possible to use the failure mode effects listed in FMEA as a guideline.

– Record the associated detection means, measurement points and functional and/or physical symptoms against the equipment FMG. If the identified detection means, measurement points, functional symptoms or physical symptoms differ between the failure modes associated with the same equipment FMG in step 1 (refer to Para 2.4), then it is necessary split into one unique equipment FMG per symptoms “signature”.
– If it is not possible to detect the associated equipment failure modes using the means above, it will be regarded as symptoms “signature” not available. For additional discussions on hidden failures, refer to S4000P.

Note

Only detection means available at workshop level are part of this analysis. As part of the equipment manufacturing process, an operational test is frequently performed to validate the manufactured equipment. This kind of test can be considered as an existing means to detect an equipment FMG, refer to Para 2.7 for detail. Para 3 describes the use of detection means built into the Product, which can also provide inputs on the detection of equipment FMG.



ICN-B6865-S3000L0149-001-01

Fig 4 Equipment failure mode grouping including failure mode symptoms

The example in Fig 4 shows that it is necessary to split the initially identified equipment **FMG 2,** into two separate equipment **FMG 2A** and equipment **FMG 2B,** due to the differences in the symptoms. This difference indicates the respective subset of the included equipment failure modes. Also note that equipment **FMG 1** and **FMG 2A** have the same symptoms and therefore require additional equipment failure mode isolation tasks in order to determine, upon noticing the symptoms signature, which of the two equipment FMG has occurred (refer to Para 2.6).

Note

The same principle also applies to sub-equipment. The only difference is the equipment failure modes defined in the example above are replaced with FMG defined for the sub-equipment (refer to example in Para 2.4).

2.6 Define equipment failure mode isolation task requirements

The next step in the equipment corrective maintenance analysis is to identify the means to isolate the equipment failure mode and define the equipment failure mode isolation task requirements, in case two or more equipment FMG have the same set of failure mode symptoms (symptoms signature). This is also referred to as symptoms ambiguity groups (refer to Para 2.5).

If the identified symptoms are unique for the equipment FMG, there is no need to define additional task requirements. However, if the symptoms identified by a Built-In Test (BIT) with a high rate of false alarms, it can be necessary to define an additional equipment FMG for an operational test task requirement (refer to Para 2.7).

For each equipment FMG that is part of a symptoms ambiguity group, the analysis must include the following steps:

1 Check if any form of test can isolate the respective equipment failure mode.

2 Check if a functional test or visual inspection can isolate the respective equipment failure mode.

3 Check if any measurement points (eg, gauges) can isolate the respective equipment failure mode.

4 Check functional symptoms and/or physical symptoms likely to help to isolate the respective equipment failure mode.

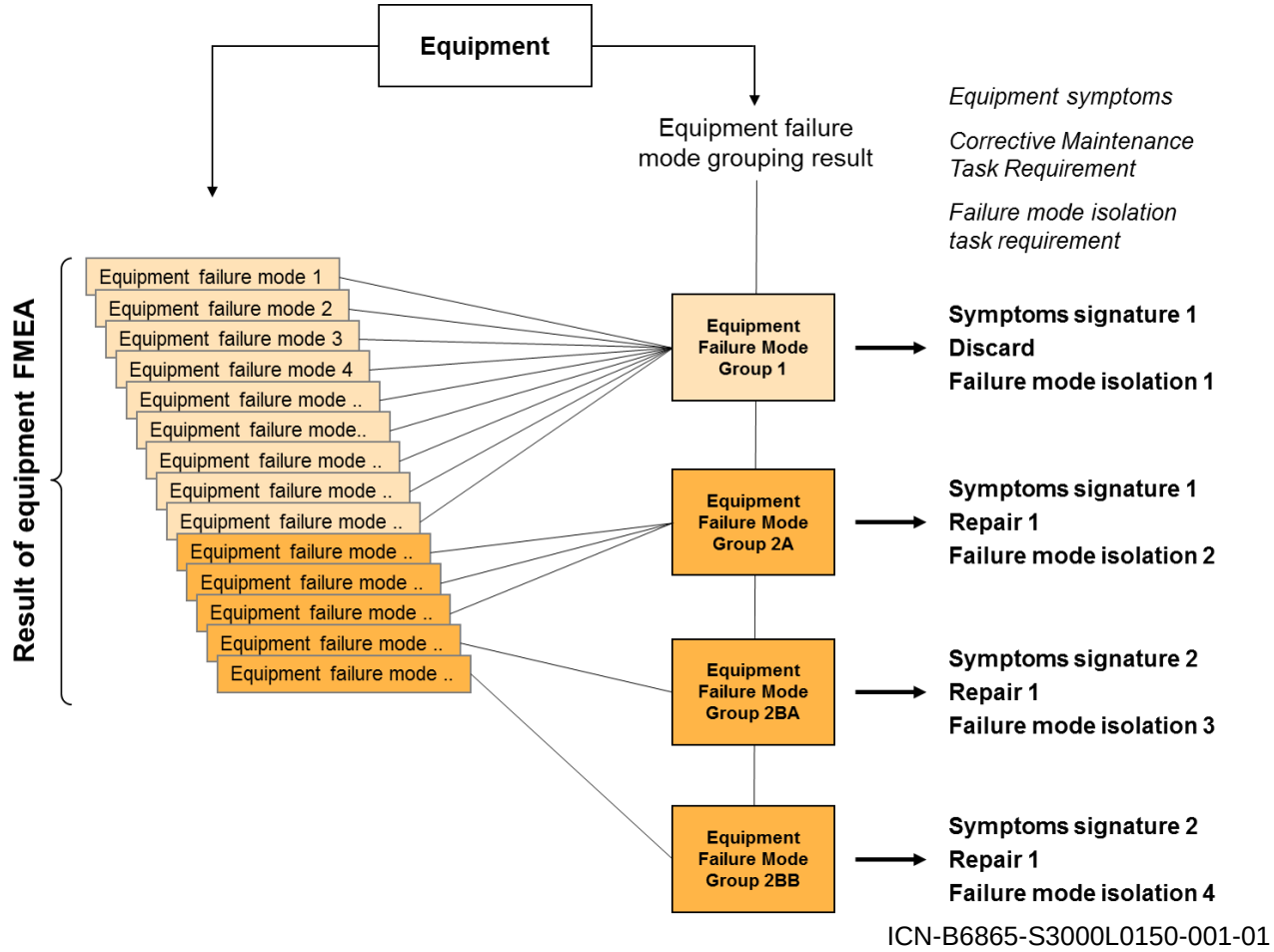
Note

Only test means available at workshop level are part of this analysis. As described in Para 2.5, it is possible to consider an existing operational test as a means to isolate an equipment FMG. Refer to Para 2.7 for details on operational test. Para 3 describes the use of test means built into the Product. In some cases, these test means can procure inputs that can be useful to isolate an equipment FMG at workshop level.

Note

The scope of the LSA does not include elaborating the equipment failure mode localization and isolation process itself (also known as troubleshooting). Although the equipment corrective maintenance analysis as defined in this chapter indicates how to isolate specific equipment failure modes, the complete troubleshooting process to prioritize and sequence failure mode isolation tasks for equipment FMG with the same symptoms signature can require additional factors (eg, probability of occurrence of each equipment FMG, elapsed time to restore each equipment FMG). Some of these factors are only available during the in-service phase (eg, status and history for serialized equipment, availability of spares parts).

The equipment failure mode isolation task requirement then must be recorded against the respective equipment FMG. If the equipment failure modes associated with the same equipment FMG in step 1 and 2 (refer to Para 2.4 and Para 2.5) have different equipment failure mode isolation task requirements, then it is possible to split the equipment FMG into a single equipment FMG per identified equipment failure mode isolation task. Even if this information is not useful at workshop level, given that the repair solution is the same for the two equipment FMG, it can provide inputs for an isolation task at Product level, if one or more equipment's FMG have the same symptoms signature.



ICN-B6865-S3000L0150-001-01

Fig 5 Equipment failure mode grouping including failure mode isolation tasks

Fig 5 shows that equipment **FMG 1** and **FMG 2A** have the same symptoms and therefore require additional equipment failure mode isolation tasks in order to verify which of the two equipment FMG has occurred upon recording the symptoms signature (refer to Para 2.5).

Furthermore, Fig 5 shows that the equipment **FMG 2B** has been split into two separate equipment FMG considering that it requires different equipment failure mode isolation tasks to determine which equipment failure mode has actually occurred. As previously said, this can provide an input for the isolation tasks at Product level in case of ambiguity between different equipment FMG.

Note

Depending on the business rules for a specific program, it is also possible to assign both **Failure mode isolation 3** and **Failure mode isolation 4** to the equipment **FMG 2B**. This indicates that if one of these failure mode isolation tasks is successful, it is necessary to perform its associated corrective maintenance task.

The documentation of equipment failure mode isolation task requirements can include references and additional descriptive information, for example:

- references to technical documents that must or can be used during the equipment failure mode isolation task (eg, technical plan, wiring diagram, interface description)
- possible crosschecks that can help to isolate the equipment failure mode. Such crosschecks are useful for identifying the sources of common causes for equipment failure modes. It is also advisable to document the symptoms to be monitored or recorded during these crosschecks.

- measurements to be carried out to check the physical and/or functional characteristics of the item under analysis, including:
  - test equipment required to perform these measurements
  - expected values for each measurement and signification of unexpected measures

There can be a requirement to assign more than one equipment failure mode isolation task to an equipment FMG. This reflects the need to step by step narrow down possible equipment failure modes that have led to the same symptoms for multiple equipment FMG. It is possible to record the logical order for performing the equipment failure mode isolation tasks in the context of the individual equipment FMG, but it is not documented in the form of logical decision trees that indicate how to prioritize the different equipment failure mode isolation paths (also referred to as troubleshooting).

Note

The same principle also applies to sub-equipment. The only difference is the equipment failure modes defined in the example above are replaced with FMG defined for the sub-equipment (refer to example in the note below Fig 4).

2.7 Define equipment operational test task requirements

At the Product level, an equipment is sent to the workshop for corrective maintenance when it has been detected as potentially at fault. At Product level, it is not always possible to identify the faulty equipment unambiguously. Sometimes, the equipment can be part of a group of equipment that has the same symptoms signature. This group of equipment is called an ambiguity group, and it is possible to replace the complete group of equipment to restore the Product's availability as fast as possible. Sometimes, the detection itself can have a degree of uncertainty (eg, a BIT on-Product level can have a high false alarm rate).

In these cases, when equipment is received in the workshop, it first undergoes an operational test to confirm the equipment is actually at fault. This kind of test can be similar to the test performed at the end of the manufacturing process for the equipment acceptance.

Altogether, this type of operational test task can meet different requirements regarding the equipment corrective maintenance analysis. The requirements are:

- Confirm equipment at fault
  As previously described, such operational test tasks can be performed on the equipment at workshop level as soon as the equipment at fault is received in the workshop. The result of the equipment operational test task will confirm that the equipment is actually at fault, or that the equipment is not at fault. In the latter case, the equipment is classified as No Fault Found (NFF) and sent back to the Product level.

- Provide an additional failure mode isolation test task
  In some cases, the operational test task can produce results that can help isolating the equipment FMG at fault, and it can be regarded as a failure mode isolation task requirement.

- Final validation test after equipment repair
  The operational test task can be performed as the last task of the repair process to validate that the equipment has been properly restored.

Regarding the equipment operational test task, the equipment corrective maintenance analysis determines the need to perform an operational test task on each equipment FMG to cover the different requirements mentioned above. Taking into account that such an equipment operational test can exist through the manufacturing process, record the equipment operational test requirement for each relevant equipment FMG.

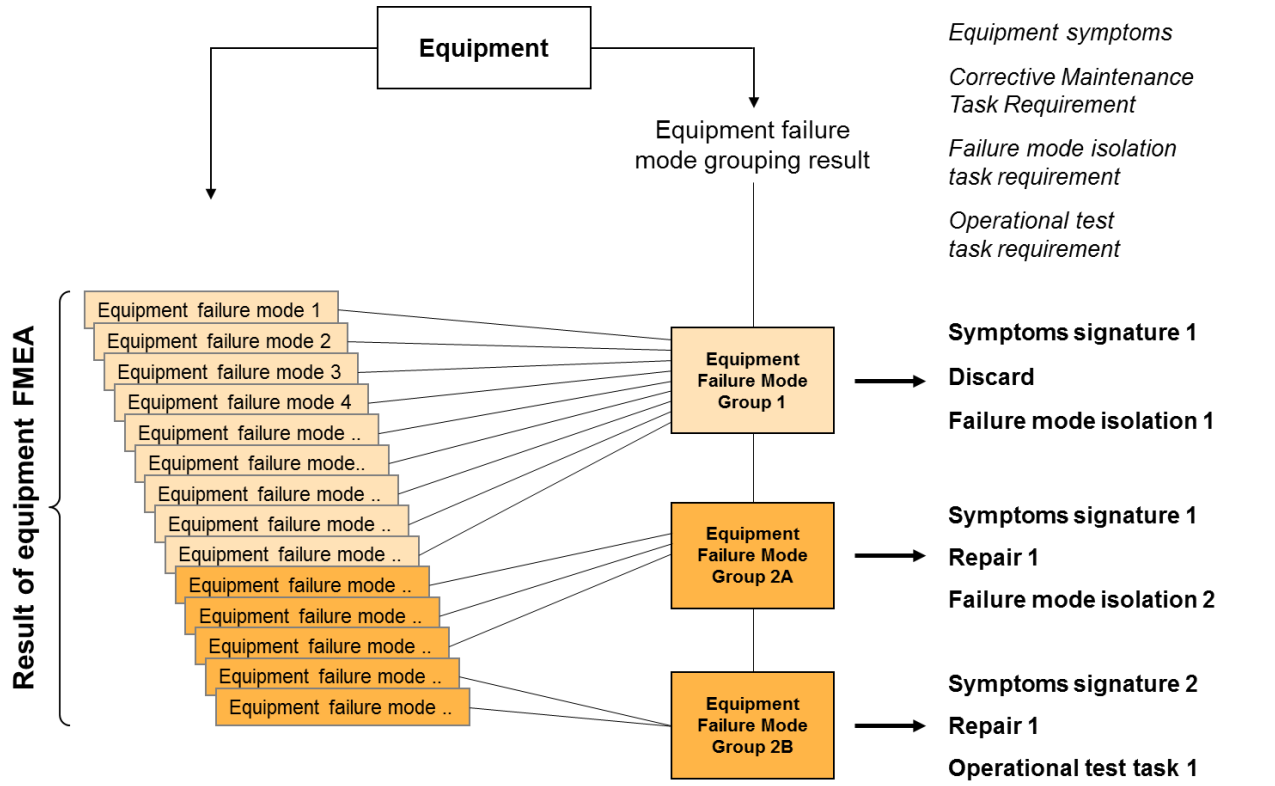
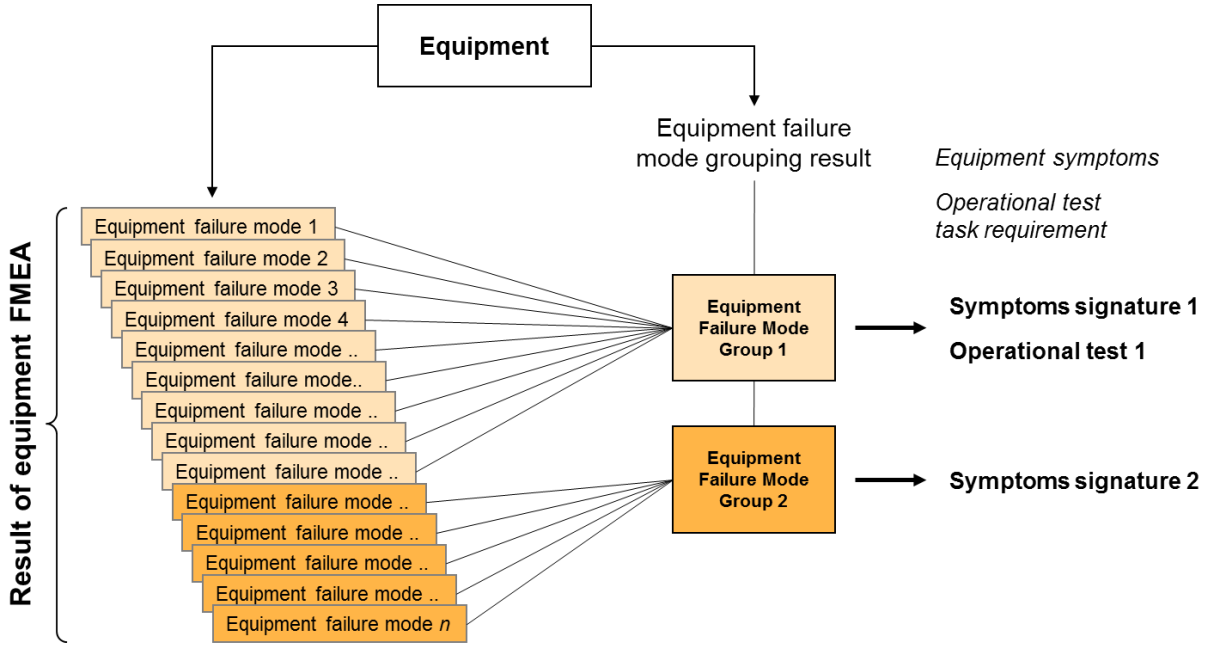


Fig 6 Equipment failure mode grouping including operational test tasks

Fig 6 shows that the equipment **FMG 2B** requires an operational test task, whereas **FMG 1** and **FMG 2A** don't. The list below shows examples where it is necessary to define requirements such as:

- if the equipment FMG symptoms signature is ambiguous (eg, high false alarm rate for one of the alarms that is part of the symptoms signature)
- if the equipment FMG symptoms signature is common to others equipment FMG, there will be an equipment ambiguity group. This can lead to having equipment sent to the workshop, with no certainty that the equipment is actually at fault.
- if the isolation task for the equipment FMG is not enough
- if there is no efficient test to validate the equipment repair action

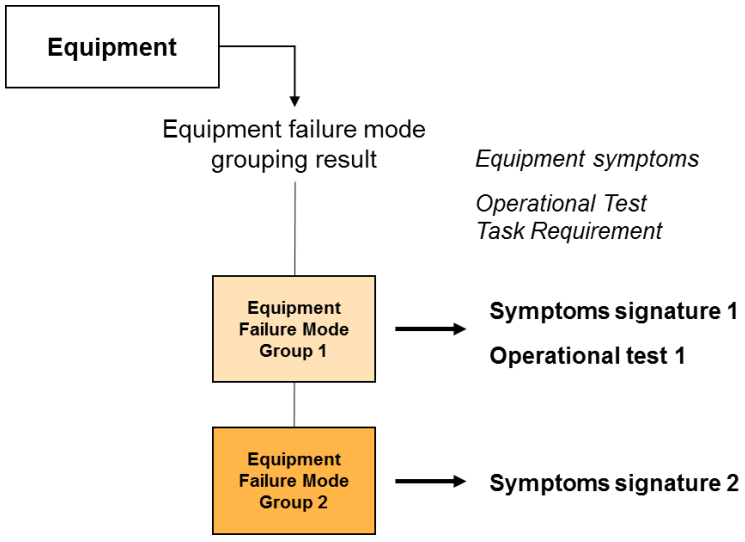
Some maintenance concepts determine that only operational tests will be performed at the customer/operator site. All other equipment corrective and failure mode isolation tasks then occur at industry level. For such maintenance concepts, the equipment corrective maintenance analysis will only consider the equipment FMG operational test task. There is no need to consider any further analysis for the respective equipment failure mode (refer to Fig 7).



ICN-B6865-S3000L0152-001-01

Fig 7 Equipment failure mode grouping only including operational test tasks

Fig 7 shows that there are two identified symptom signatures, one of which requires an operational test before the equipment is sent to industry for further repair or discard activities. If the operational test for **Symptoms signature 1** is successful, it is possible to record it as NFF. The example in Fig 7 refers back to a set of equipment failure modes that is the basis for the respective symptoms signature. This is not an absolute requirement if the OEM/supplier only defines symptoms and test tasks as a simplified approach (refer to Fig 8).



ICN-B6865-S3000L0153-002-01

Fig 8 Equipment failure mode grouping only including operational test tasks (simplified)

Note

The same principle also applies to sub-equipment. The only difference is the equipment failure modes defined in the example above are replaced with FMG defined for the sub-equipment (refer to the example in the note below Fig 4).

2.8 Inputs

The following inputs are required to perform the equipment corrective maintenance analysis:

- Candidate Item List (CIL)
- equipment part list (often referred to as EBOM)
- equipment FMEA (often referred to as technical FMEA)
- operational test task used for equipment acceptance in the manufacturing process

2.9 Outputs

The equipment corrective maintenance analysis produces the following output:

- equipment FMG and its:
  - associated equipment failure modes
  - failure mode symptoms
  - corrective maintenance task requirements
  - failure mode isolation task requirements
  - equipment operational test task requirements

3 On-Product corrective maintenance

3.1 Introduction

The on-Product corrective maintenance analysis aims to define on-Product corrective maintenance task requirements, and to identify a method to detect and isolate Product failure modes.

The result is a set of Product FMG, where each FMG is a unique combination of:

- one corrective maintenance task requirement, which is common to the included (grouped) Product failure modes (ie, all the included Product failure modes lead to the exact same corrective action)
- one or more symptoms that are common to all Product failure modes grouped against the Product FMG
- one or more failure mode isolation task requirement that can identify the included Product failure mode that has occurred

Note

There can be more than one defined failure mode isolation task requirement for any Product FMG. This is required whenever it is not possible to determine directly and without ambiguity which of the failure modes associated with the Product FMG has occurred. In this case, it is necessary to gradually reduce the number of possible Product FMG, in order to eventually isolate one Product FMG.

As described at Para 2, corrective maintenance analysis starts analyzing each FMG defined for the equipment to be installed on the Product, using a bottom up approach and considering the equipment in the context of the Product. This step is a grouping activity where the defined equipment FMG are analyzed from the perspective of what corrective maintenance task is required to repair/restore the Product.

The next step is to analyze and group the failure modes defined for the structure (eg, fuselage) and the installation details (eg, wires, tubes, fasteners). In the same way as for equipment corrective maintenance analysis, the on-Product corrective maintenance analysis will produce a set of FMG. The difference is that Product FMG are primarily associated with breakdown elements representing, for example, installation locations, families of hardware items (eg, composites, tubes, wiring). Refer to Chap 4.

It is then necessary to analyze each Product FMG on the basis of the identified symptoms, which can support the detection/isolation of individual failure modes. Symptoms can be built-in

test capabilities, measurement points and/or local or next higher failure mode effects as recorded in the system/functional FMEA/FMECA defined for the Product.

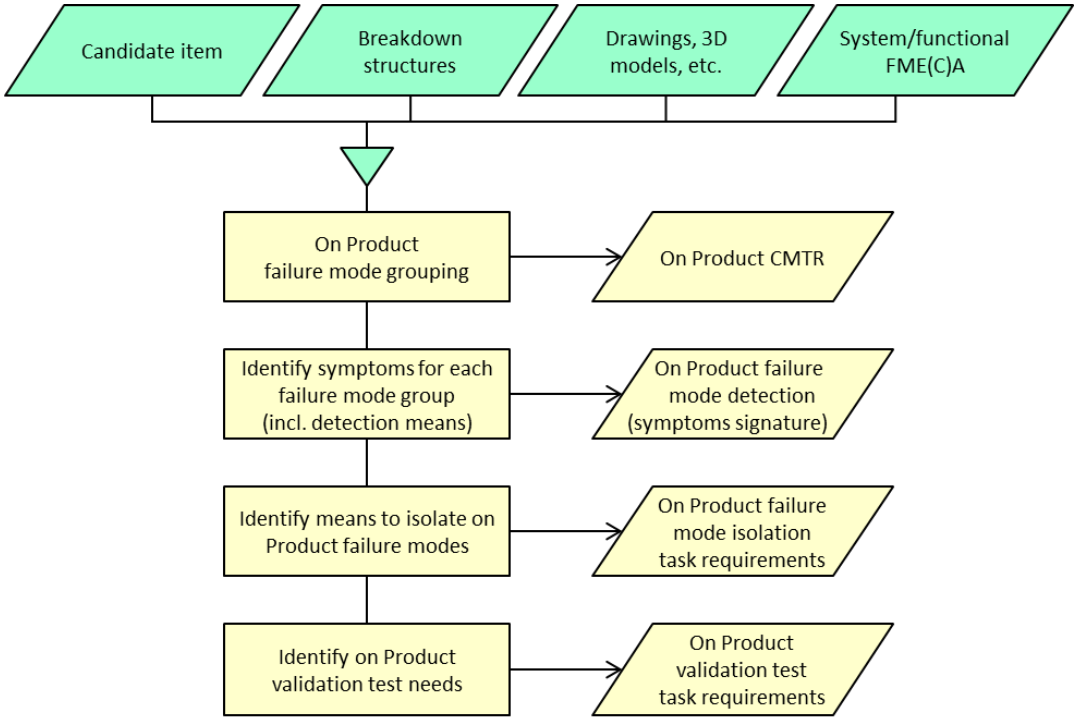
If the symptoms defined for a Product FMG are ambiguous (ie, different Product FMG have the same symptoms signature), it is necessary to associate the Product FMG with a failure mode isolation task requirement.

Note

Product corrective maintenance analysis must consider available failure mode detection means incorporated into the Product.

3.2 Method

Fig 9 shows the method for the on-Product corrective maintenance analysis in the form of a flow chart.



ICN-B6865-S3000L0154-001-01

Fig 9 Logical flowchart for on-Product corrective maintenance analysis

The method for performing on-Product corrective maintenance analysis includes the following sub-processes:

1 Perform on-Product failure mode grouping and define corrective maintenance task requirements (refer to Para 3.4)

2 Identify the means (symptoms) for on-Product failure mode detection (refer to Para 3.5)

3 Identify the means for on-Product failure mode isolation and define on-Product failure mode isolation task requirements (refer to Para 3.6)

4 Identify the need for Product validation tests (refer to Para 3.7)

Note

The on-Product corrective maintenance analysis is an iterative process.

3.3 Candidate items

The first activity is to identify the installation locations (equipment), structure and installation details that will undergo the on-Product corrective maintenance analysis. This activity is part of the overall candidate selection activity described in Chap 3.

Examples of specific questions related to the selection of candidate items for on-Product corrective maintenance analysis are:

- Is there equipment likely to fail during the estimated Product operational life?
- Are there any structure components that are likely to degrade beyond acceptable limits during the estimated Product operational life?
- Are there any installation details (or families of installation details) that are likely to fail during the estimated Product operational life?

Whether equipment, structural component or installation detail is subject to failure or degrading, is often based on an analysis which include criticality assessment, available redundancies etc. Refer to S4000P.

3.4 On-Product failure mode grouping

3.4.1 Possible on-Product failure mode causes

Based on the different types of corrective maintenance candidate items described in Para 3.3, this paragraph includes five major failure mode grouping activities. These are:

- installed equipment failure mode grouping
- family based installation and structure failure mode grouping
- equipment installation details failure mode grouping
- general system/function failure mode grouping
- general structure failure mode grouping

3.4.2 Installed equipment failure mode grouping

Installed equipment failure mode grouping is a bottom-up approach where the equipment FMG is analyzed in the context of its respective installation location. Installation locations are best represented as hardware elements in a Product breakdown structure (refer to Chap 4).

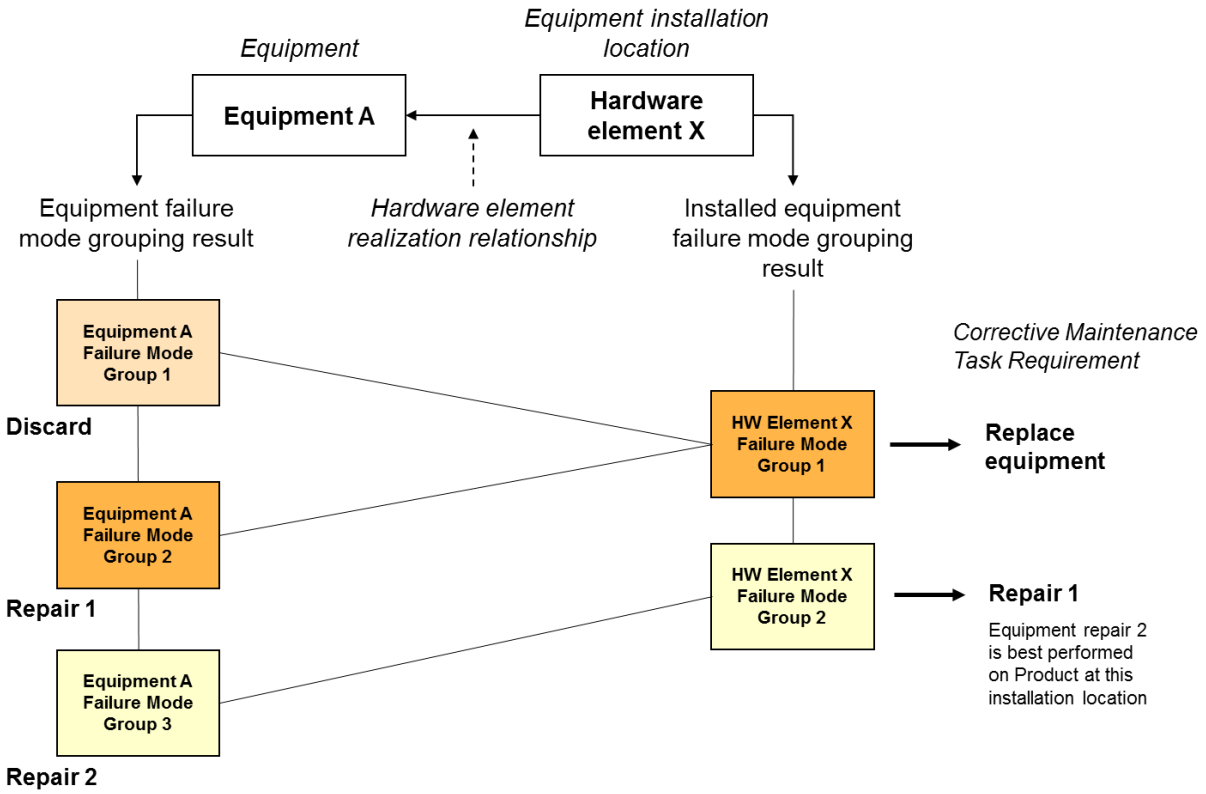
The rationale for this approach is to gradually determine the Product FMG and its associated corrective maintenance task requirements for each on-Product installation location, and to ensure that the equipment FMG is analyzed when put into the context of individual installation locations (refer to Fig 10).

Note

On-Product failure mode grouping does not need to consider what will be done with the equipment, its sub-equipment and components once removed from the Product. Sometimes, however, on-Product FMG can already point to specific maintenance activities to be performed on shop level after removal of the faulty equipment.

Note

The on-Product failure mode grouping analysis must only consider the FMG defined for the equipment to be installed.



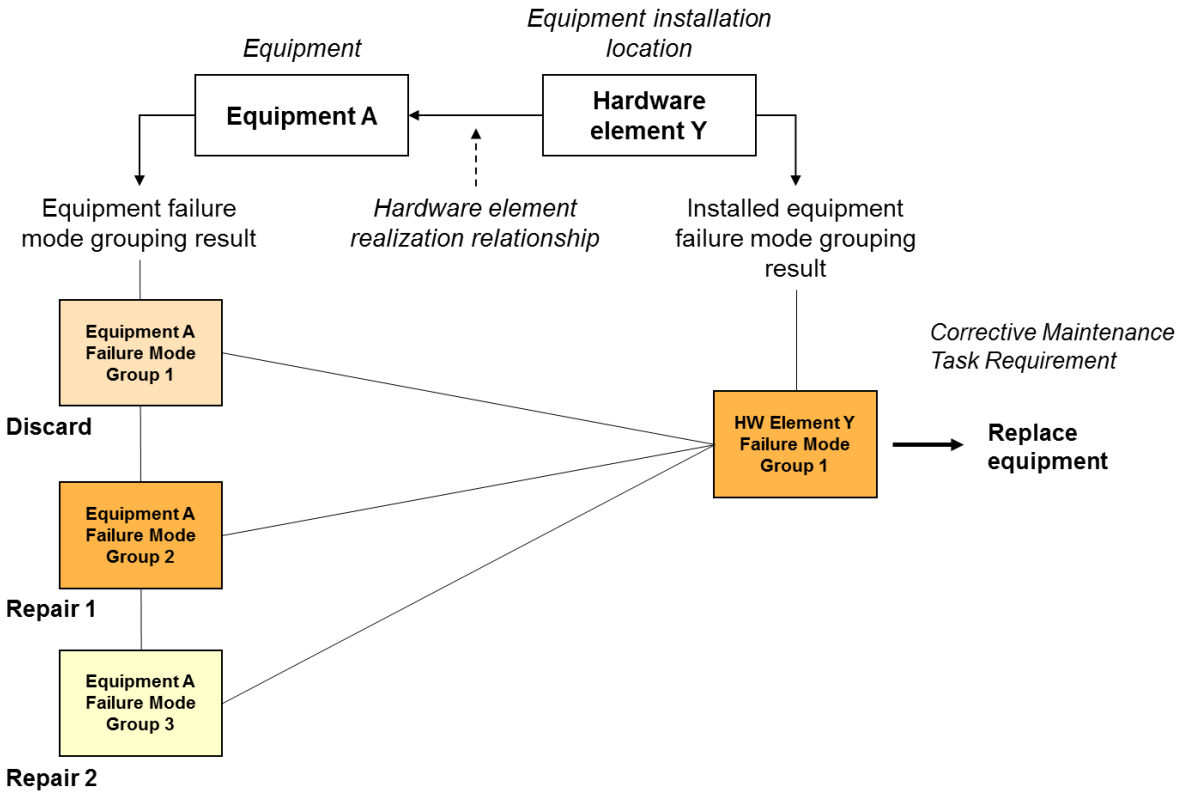
ICN-B6865-S3000L0155-001-01

Fig 10 Installed equipment failure mode grouping (1)

Note

As shown in Fig 10, the corrective maintenance task requirement does not indicate the name of the equipment to be replaced on-Product because there can be more than one type of equipment installed at the installation location represented by **Hardware element X**.

Equipment installed at different locations can result in totally different FMG, for example due to access aspects (refer to Fig 10 and Fig 11). Fig 11 shows that it is not possible to repair the equipment at Product level for **Hardware Element Y** (eg, due to tool access).



ICN-B6865-S3000L0156-001-01

Fig 11 Installed equipment failure mode grouping (2)

Other situations to consider when performing the installed equipment failure mode grouping are:

- more than one equipment can be installed at a given location (form, fit and function equivalent)
- more than one equipment can be installed at a given location (form, fit and function equivalent), but they can have differences in their respective FMG (eg, one equipment has a battery that can be replaced while another equipment does not)
- totally different equipment can be installed (eg, for mission configurations)

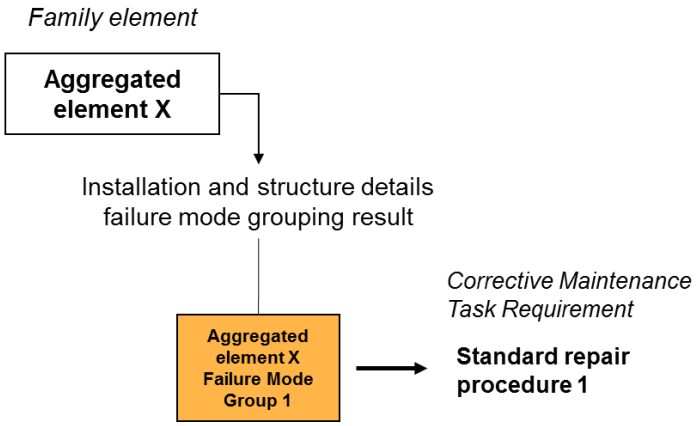
Each of these examples can have an impact on both Product breakdown structures as well as on scoping for individual FMG. It is necessary to define detailed decisions for each project.

3.4.3 Family based installation and structure components failure mode grouping

The installation and structure components failure mode grouping covers two major areas.

- installation details that serve one or many pieces of equipment, and/or one or many systems/functions (eg, electric wiring, hydraulics)
- structure components which support the entire Product

Different types of platforms can have different approaches for performing on-Product corrective maintenance analysis and failure mode grouping, but the common denominator should be an FMG that reflects standard repair procedures for the type of installation and structure technology used (refer to the descriptions of breakdown elements that represent families of parts, Chap 4). Fig 12 shows the basic principle.



ICN-B6865-S3000L0157-002-01

Fig 12 Family-based failure mode grouping

Note

The family concept described in Chap 4 does not consider parts and part identifiers.

Note

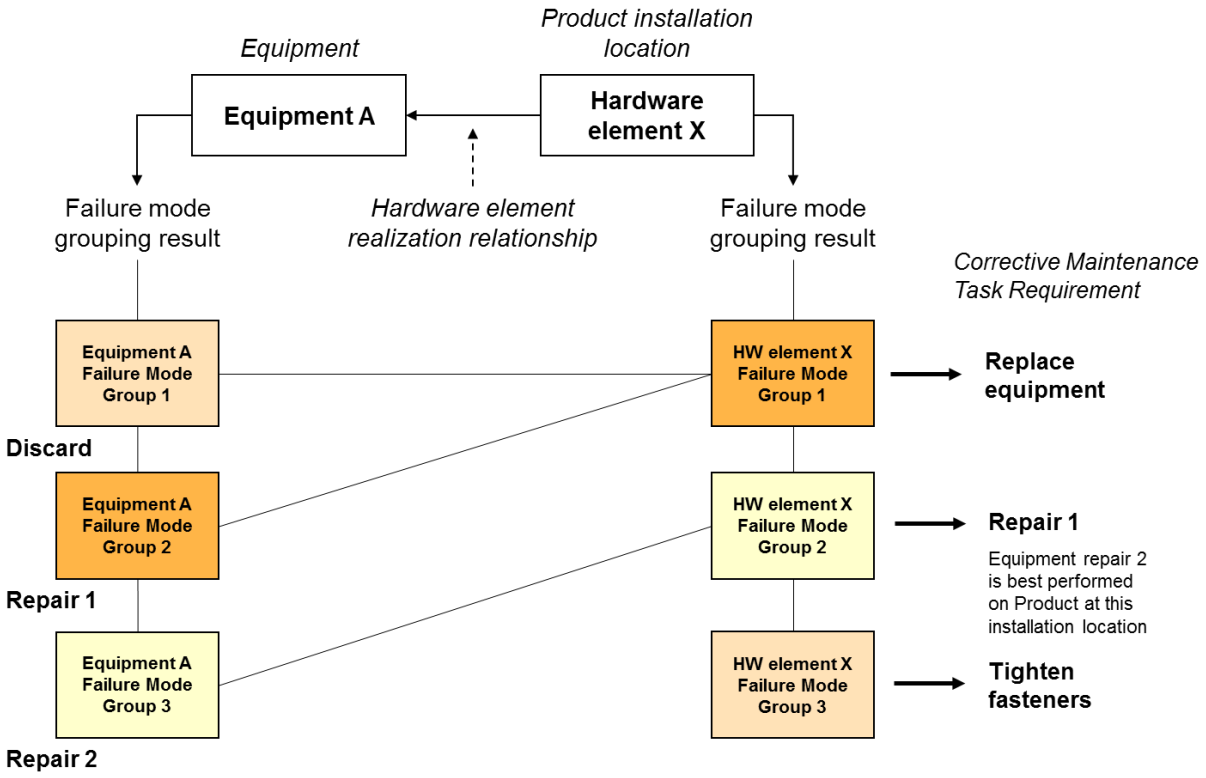
In most cases, no FMEA is performed at this level of detail.

3.4.4 Equipment installation details failure mode grouping

The equipment installation details failure mode grouping considers installation details that are directly associated with the installation of single pieces of equipment (eg, fasteners, connectors).

The equipment installation details failure mode grouping is an extension of the installed equipment failure mode grouping (refer to Para 3.4.1), and it takes into consideration installation details failure mode grouping based on family (refer to Para 3.4.2). The rationale for performing this analysis after carrying out the installation details failure mode grouping based on family is that it will probably not be necessary to repeat the general corrective task requirement for individual installation locations. In many cases, however, there will only be corrective maintenance task requirements defined specifically for individual installation locations (eg, for failure modes where fasteners must be tightened because of vibrations). This is a failure mode effect. Refer to Fig 13.

System/functional FMEA/FMECA can identify equipment installation details failure modes, but it is not always the case. Therefore, it is advisable to always analyze possible installation details failure modes for each installation location as part of the on-Product corrective maintenance analysis.



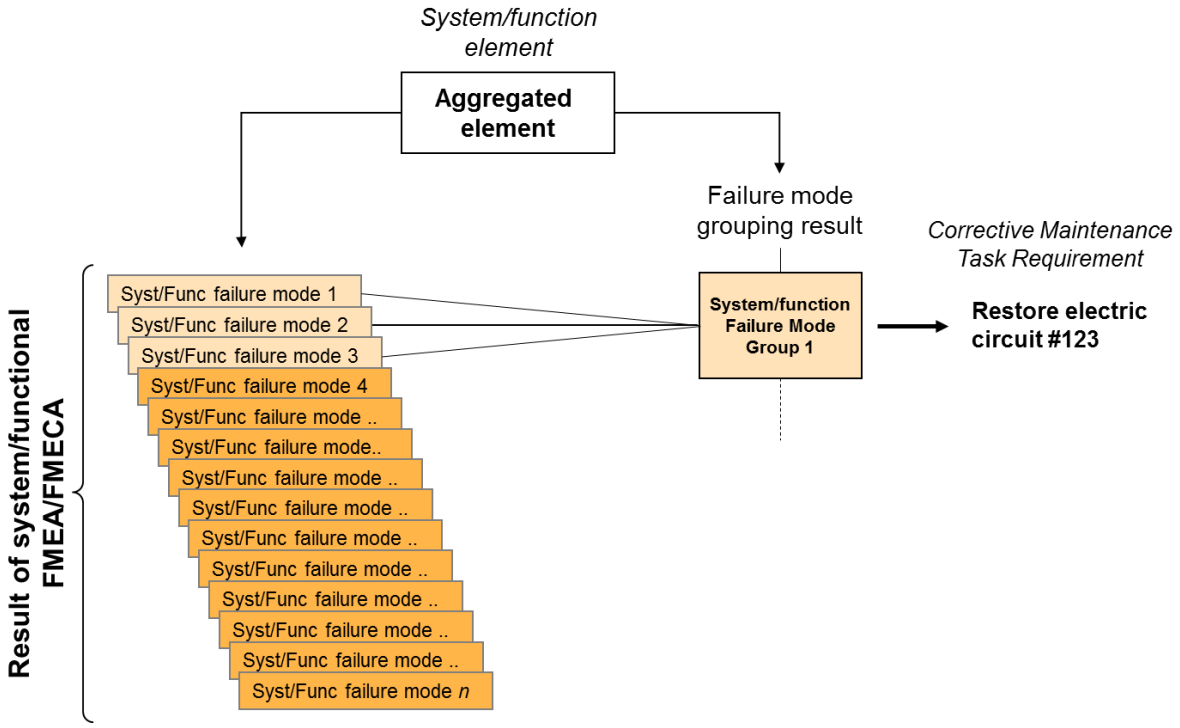
ICN-B6865-S3000L0158-001-01

Fig 13 Equipment installation details failure mode grouping

3.4.5 General system/function failure mode grouping

The general system/function failure mode grouping is often organized against the respective system/function. The rationale for performing this analysis after carrying out the installation details failure mode grouping based on family (refer to Para 3.4.2) is that it will probably not be necessary to repeat the standard corrective task requirement for individual systems/functions. For example, there can be a system/function-oriented on-Product FMG, which does not have an associated corrective maintenance task requirement, but only a set of symptoms and failure mode isolation task requirements.

However, in many cases there can also be uniquely defined corrective maintenance task requirements for the system/function on-Product (eg, restore an electric circuit, refer to Fig 14). System/functional FMEA/FMECA can identify system/function failure modes, but it is not always the case. Therefore, it is advisable to always analyze possible general system/function failure modes as part of the on-Product corrective maintenance analysis.



ICN-B6865-S3000L0159-001-01

Fig 14 General system/function failure mode grouping

Note

General system/function failure modes can often be detected by BIT.

3.4.6 General structure failure mode grouping

General structure failure mode grouping is probably not as common as other failure mode grouping analysis activities. In many cases, special events and damages will call for corrective maintenance on the general structure (refer to Chap 8). However, it is worth mentioning general structure failure mode grouping as an explicit analysis activity.

When performed, it will be very similar to a general system/function failure mode grouping, with the following exceptions:

- The on-Product FMG for structure failure modes is defined for a breakdown element that represents the structure or portion of the structure
- There will be a lower degree of built in test capabilities

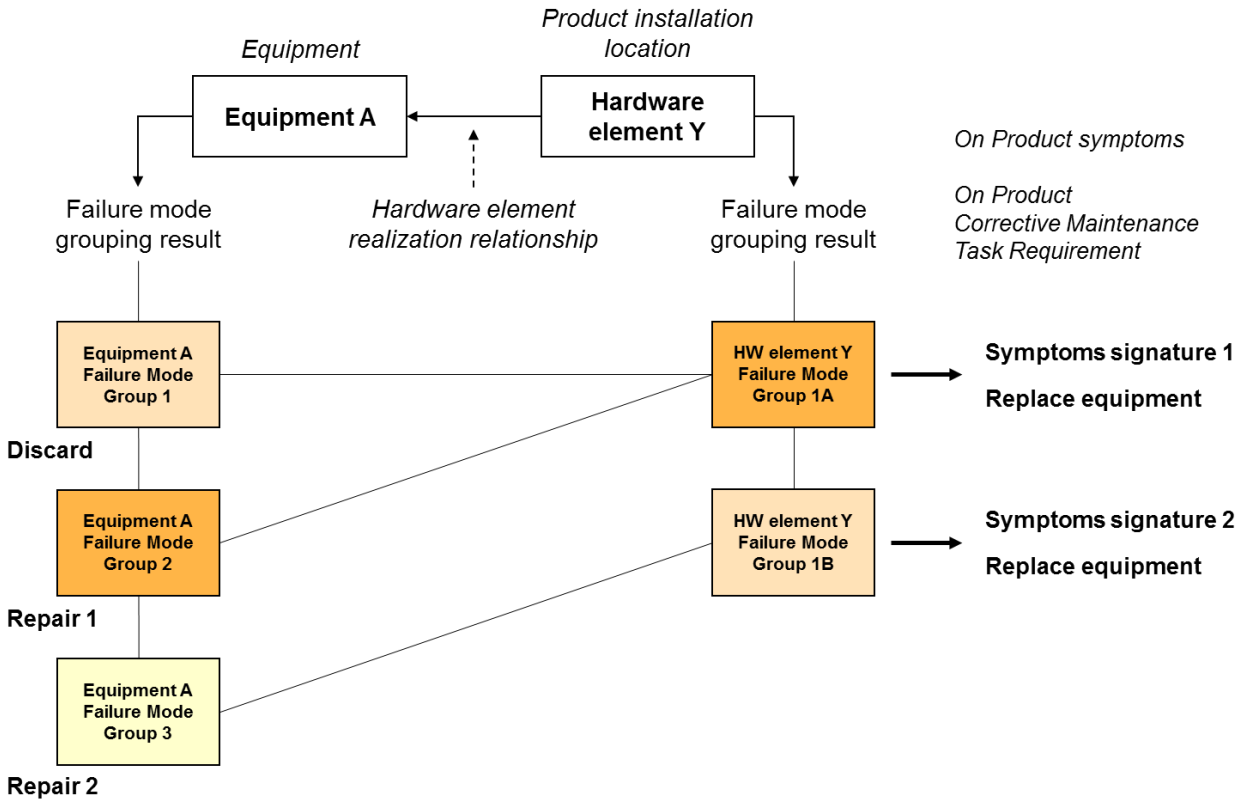
3.5 Identify means (symptoms) for on-Product failure mode detection

The identification of on-Product failure mode detection means (symptoms) follows the same logic as described for equipment (refer to Para 2.5). It is necessary to analyze each possible on-Product failure mode (including equipment FMG) considering:

- checking whether there is any form of automatic detection that triggers a warning device whenever an on-Product failure mode occurs. If yes, assess the detection rate and false alarm rate of this automatic detection mean.
- checking whether there are any measurement points (eg, gauges) that can indicate that the on-Product failure mode has occurred
- checking functional symptoms and/or physical symptoms that can help detect the on-Product failure mode. It is possible to use the system/function failure mode effects listed in the system/functional FMEA/FMECA as a guideline.

- recording the associated detection means, measurement points and functional and/or physical symptoms against the on-Product FMG. If the identified detection means, measurement points, functional symptoms or physical symptoms differ between the on-Product failure modes associated with the same on-Product FMG in step 1 (refer to Para 3.4), then it is necessary to create a single on-Product FMG for each symptoms "signature".
- If it is not possible to detect the on-Product failure mode using the means above, it will be regarded as symptoms "signature" not available. For additional discussions on hidden failures, refer to S4000P.

Note
The analysis must only include detection means that are available at Product level. Para 2 describes the use of detection means for equipment on bench.



ICN-B6865-S3000L0160-001-01

Fig 15 On-Product failure mode grouping, including failure mode symptoms

Fig 15 shows that the initially identified on-Product **FMG 1** has been split into **FMG 1A** and **FMG 1B** due to the differences in symptoms. Each division indicates the respective subset of the included equipment FMG.

3.6 Define on-Product failure mode isolation task requirements

The identification of on-Product failure mode isolation task requirements basically follows the same logic as described for equipment (refer to Para 2.6).

If the identified symptoms are unique for the on-Product FMG, there is no need to define additional on-Product failure mode isolation task requirements. However, if the symptoms identified by a BIT with a high rate of false alarms, it can be necessary to define an additional on-Product FMG test task (also refer to Para 2.7).

For each on-Product FMG that is part of a symptoms ambiguity group, the analysis must include the following steps:

1 Check if any form of test can isolate the respective on-Product failure mode

2 Check if a functional test or visual inspection can isolate the respective on-Product failure mode

3 Check functional symptoms and/or physical symptoms likely to help to isolate the respective on-Product failure mode.

Note

Only test equipment used for testing on Product level are part of this analysis.

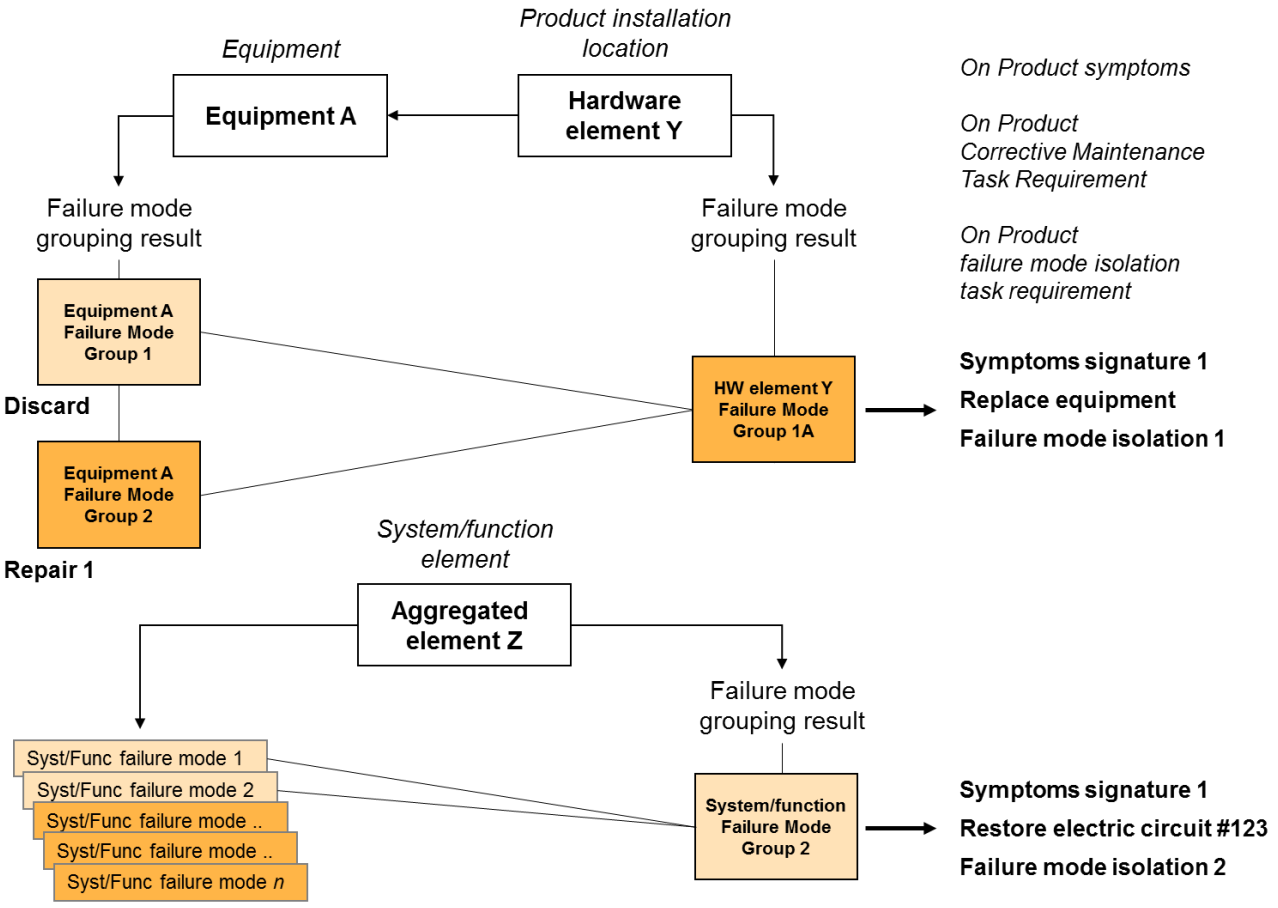
Note

The scope of LSA does not include elaborating the on-Product failure mode localization and isolation process itself (also known as troubleshooting). Although the on-Product corrective maintenance analysis as defined in this chapter indicates how to isolate specific Product failure modes, the complete troubleshooting process to prioritize and sequence failure mode isolation tasks for Product FMG with the same symptoms signature can require additional factors (eg, probability of occurrence of each Product FMG, elapsed time to restore each Product FMG). Some of these factors are only available during the in-service phase (eg, status and history for serialized equipment, availability of spares parts).

Note

Symptom ambiguity groups can occur across the different analysis steps described in Para 3.4. This means that an on-Product FMG identified as part of the equipment installation failure mode grouping (refer to Para 3.4.1) can have the same symptoms as an on-Product FMG identified as part of general installation details failure mode grouping (refer to Para 3.4.4).

Afterwards, it is necessary to record the on-Product failure mode isolation task requirement against the respective FMG. If the on-Product failure modes associated with the same FMG in step 1 and 2 (refer to Para 3.4 and Para 3.5) have different Product failure mode isolation task requirements), then it is necessary to split the on-Product FMG into a single FMG for each identified on-Product failure mode isolation task.



ICN-B6865-S3000L0161-001-01

Fig 16 On-Product failure mode grouping, including failure mode isolation tasks

Fig 16 shows that the on-Product **FMG 1A** defined for **Hardware element Y** and the on-Product **FMG 2** defined for system/function **Aggregated element Z** have the same symptoms signature, and therefore require additional on-Product failure mode isolation tasks.

The documentation of on-Product failure mode isolation task requirements can include references and additional descriptive information, for example:

- references to technical documents that must or can be used during the on-Product failure mode isolation task (eg, technical plan, wiring diagram, interface description)
- possible crosschecks that can help to isolate the on-Product failure mode. Such crosschecks are useful for identifying the sources of common causes for on-Product failure modes. It is also advisable to document the symptoms to be monitored or recorded during these crosschecks.
- measurements to be carried out to check the physical and/or functional characteristics of the item under analysis, including:
  - test equipment required to perform these measurements
  - expected values for each measurement, and signification of unexpected measures

There can be a requirement to assign more than one on-Product failure mode isolation task to one on-Product FMG. This translates into the need to narrow down the possible on-Product failure modes that have led to the same symptoms for multiple on-Product FMG.

3.7 Validation test

At Product level, it is not always possible to identify the faulty component (installed equipment, equipment installation, general installation detail, etc.) unambiguously. Sometimes, the component can be part of a group of components that have the same set of symptoms. This group of components is called an ambiguity group. Depending on the circumstances, (eg, requirement to restore Product's availability as fast as possible), the corrective maintenance tasks are applied even if the component actually at fault has not been definitely identified. Sometimes, the detection can have a degree of uncertainty (eg, a BIT on-Product level can have a high false alarm rate). Sometimes, the isolation test tasks possible at Product level are not good enough.

In these cases, there is a need to confirm the Product has been successfully repaired. This action is often called validation test task, and it consists in verifying that all symptoms have disappeared. There are different types of validation test task requirements, for example a BIT, a measurement, a functional test or an inspection. Regarding validation tests, the on-Product corrective maintenance task analysis consists in identifying all means available for each on-Product FMG to ensure that the symptoms have disappeared. During the repair process of the Product, a validation test will be defined using an optimized combination of means corresponding to all on-Product FMG. It is necessary to document the corresponding Product validation test requirement.

3.8 Inputs

The following inputs are required in order to perform the on-Product corrective maintenance analysis:

- CIL
- Product breakdown structures
- drawings, 3D models etc.
- system/functional FMEA
- equipment corrective maintenance analysis results for each Product equipment and the relevant FMG symptoms

3.9 Outputs

The following output will be recorded during equipment corrective maintenance analysis:

- On-Product FMG and their:
  - associated equipment FMG
  - associated installation details and structure failure modes
  - symptoms
  - corrective maintenance task requirements
  - failure mode isolation task requirements
  - Product validation test task requirements

4 Relevant elements of the S3000L data model

The following Units of Functionality (UoF) support the documentation of the data associated to this chapter. Refer to Chap 19:

- S3000L UoF Environment Definition
- S3000L UoF LSA Candidate
- S3000L UoF Failure Mode
- S3000L UoF LSA Failure Mode Group
- S3000L UoF Failure Mode Symptom
- S3000L UoF Failure Mode Isolation
- S3000L UoF Task Requirement

Chapter 8

Damage and special event analysis

Table of contents

List of tables

List of figures

References

Table 1 References

Chap No./Document No.	Title
Chap 3	LSA process
Chap 4	Product structures and change management in LSA
Chap 12	Task requirements and maintenance task analysis
Chap 19	Data model
S4000P	International specification for developing and continuously improving preventive maintenance

1 General

The overall maintenance concept must include any maintenance activity that must be performed after damage or special events occur during normal Product operation.

1.1 Purpose

This chapter aims at providing a method for the identification and justification of maintenance tasks needed after damage or a special event. The analysis process described in this chapter derives as much as possible from the analytic process included in S4000P.

Considering that damage and special events can also occur on a new Product, in-service feedback on similar Products can help identifying appropriate maintenance task requirements.

1.2 Scope

Special events and damage can occur throughout the service life of a Product. This analysis process aims to anticipate the required maintenance task to rectify the special event or the damage.

1.3 Terms, abbreviations and acronyms

Table 2 provides the definitions of the terms that are specific to damage and special event analysis.

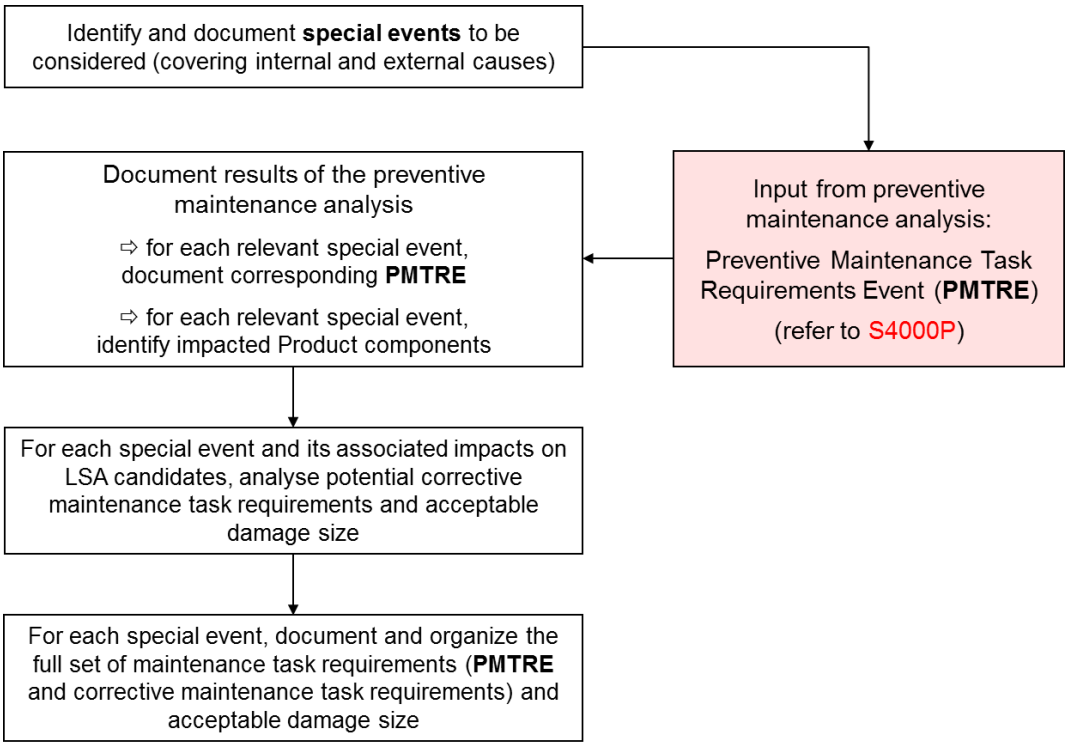
Table 2 Terms, abbreviations and acronyms

Term	Definition
Damage	A loss or reduction of functionality, excluding inherent failure (intrinsic reliabilities). It normally requires a maintenance task. It is possible to group different types of damage into families. For example, scratches, dents and cracks are typical structural damage. These damage families are candidates for standard repair procedures.
External cause	A cause is external when an event independent of Product usage occurs.
Internal cause	A cause is internal when it is a result of Product use.
Special event	A special event occurs during a Product's life and cannot be considered as normal operation. It can be due either to external causes (eg, meteorological phenomenon) or due to abnormal use (eg, over-G maneuver of an aircraft).

2 Special event analysis

2.1 Special event analysis logic

Special event analysis identifies maintenance task requirements that are justified by the occurrence of special events. Refer to Fig 1.



ICN-B6865-S3000L0068-003-01

Fig 1 General flow chart of special event analysis logic

2.2 Special event analysis process

The special event analysis process involves the following activities:

– the identification of relevant special events, refer to Para 2.2.1

- the identification of systems, subsystems, equipment, structural items and zones that the special event can affect, refer to Para 2.2.2
- a description of the impact of a special event and identification of the Preventive Maintenance Task Requirements Event (PMTRE), refer to Para 2.2.3
- the identification of the potential corrective maintenance task requirements, including data on acceptable level of damage, refer to Para 2.2.4
- the creation of the full set of PMTRE and corrective maintenance task requirements and acceptable level of damage for each special event, refer to Para 2.2.5

2.2.1 Identification of special events

The basis for this analysis process is a simple Product usage analysis. It identifies the relevant special events and their specific causes (refer to Table 3) in the context of the Product usage phases. This process follows five steps:

Step 1: List the different causes that can produce a special event.

Special events can occur due to external or internal causes, and other elements can affect them, including natural phenomena or humans. Table 3 provides examples of different types of causes. It is recommended to develop a specific table for each project to identify the relevant special events for the Product in use and for the operational environment.

Table 3 Examples of causes of special events

	Types of causes	Examples
External causes	Natural phenomena	– Meteorological (eg, lightning strike, hail) – Animal (eg, bird strike) – Stones, trees, etc.
	Caused by humans	– Threat of combat – Material maneuver
	Caused by operating environment	– Electromagnetic field – Salt, sand or pollution laden atmosphere
	Caused by transport and storage conditions (eg, sea, air, truck, rail)	– Shocks, movements and vibrations – Humidity – Depressurization
Internal causes	Caused by misuse	– Over operational limits like "hard landing", "over-G maneuver"
	Caused by internal dysfunction	– Excessive heat – Excessive pressure – Excessive vibration

Step 2: Identify the Product usage phases.

Product usage phases are, for example, operation, maintenance, storage or transport. If necessary, it is possible to divide usage phases further into sub-phases. For example, the usage phase for an aircraft can be subdivided into the sub-phases take off, cruise flight and landing.

Step 3: Apply the type of causes to the different Product usage phases to identify all possible special events.

Examples of special events due to an external cause include, but are not limited to:

- impacts on external skin due to cargo handling equipment during transportation preparation
- multiple impacts due to hail during outside parking of a vehicle
- water entrapment caused by heavy rain due to a leak during outside parking of a vehicle
- impacts on inner external skin due to a debris on a runway during take off
- lightning strike on an aircraft during cruise flight or parking

Examples of special events due to an internal cause include, but are not limited to:

- excessive stress on a tank structure due to dysfunction of the pressure relief valve during a refueling operation on ground
- excessive temperature in a bay in case of warm air leak during operation
- over-G maneuver of an aircraft during a training flight
- an unexpected wear of mechanical parts during operation

Step 4: Analyze each possible special event in order to determine its probability of occurrence.

It is possible to apply a quantitative approach based on feedback from previous similar Product usage scenarios or by statistics (eg, the average or actual number of bird strikes happening every year at a specific airport).

If it is not possible to obtain sufficient statistics, it is possible to apply a qualitative approach (refer to Table 4) or a simple description. For this approach, take into account the hypothesis defined by establishing product usage data within Chap 3.

Table 4 Example of probability rating of occurrence of special events

Rating	Occurrence	Description
1	Extremely unlikely	A special event whose probability of occurrence is essentially zero
2	Remote likelihood	A special event whose probability of occurrence is unlikely. Rare numbers of special events are likely to happen.
3	Occasional	A special event whose probability of occurrence is occasional. A few special events are likely to happen.
4	Reasonably probable	A special event whose probability of occurrence is moderate. A certain number of special events are likely to happen (to be determined project specific)
5	Frequent	A special event whose probability of occurrence is very high. This special event is almost certain to occur.

Step 5: Depending on the occurrence rating from Step 4, it is necessary to determine whether the special event is relevant for further analysis. If so, refer to Para 2.2.2. If not, document the decision. It is possible to use the rating level to determine a threshold for tailoring analysis activities.

2.2.2 Identification of impacted LSA candidates

The following questions determine the impact on the Product for each relevant special event:

- Which systems/subsystems does the special event under analysis affect?
- Which structural components of the Product does the special event under analysis affect?
- Which equipment does the special event under analysis affect?
- Which Product zones does the special event under analysis affect?

The results associated to each special event provide input for more detailed analysis to determine the impact of the special event and the PMTRE. Refer to Para 2.2.3.

2.2.3 Identification of special event impacts and PMTRE

All relevant special events are analyzed in detail to identify their impact and to determine an applicable and effective PMTRE (refer to S4000P).

For each relevant special event, the LSA data must document the following aspects:

- For each affected system or subsystem: Description of expected impacts on complete systems/subsystems, and identification of applicable and effective PMTRE.
- For each affected Product structure: Description of expected impacts on Product structure (including structural items, Structure Significant Items (SSI) and Significant Details (SD)), and identification of applicable and effective PMTRE.
- For each affected piece of equipment: Description of expected impacts on equipment (considering their specific installation locations), and identification of applicable and effective PMTRE.
- For each affected zone: Description of expected impacts on complete zones, and identification of applicable and effective PMTRE.

Note

If the special event analysis method following S4000P is not applicable to a project, it is necessary to identify another method.

2.2.4 Identification of acceptable damage and corrective maintenance task

For all relevant special events, determine a possible corrective maintenance task requirements and a corresponding acceptable level of damage. This holds for all impacted LSA candidates, regardless of whether they are systems/subsystems, a Product structure or a piece of equipment. The corrective maintenance task requirements identified for the corresponding LSA candidates can justify a standard repair or a specific damage repair task.

Note

The following aspects influence the identification of an acceptable level of damage:

- the damage does not exceed an acceptable limit ⇨ no need to repair the item immediately
- the damage exceeds an acceptable limit and needs a corresponding repair (standard or specific)

2.2.5 Full set of information for special events

Finally, it is necessary to document the full set of data/information for each special event. This includes:

- impacted LSA candidates and corresponding impacts
- identified PMTRE
- identified corrective maintenance task requirements
- information on acceptable damage limits

For each special event, the full set of information is the input for the MTA, as described in Chap 12.

2.3 Special event analysis process - inputs

The special event analysis requires a set of inputs, which includes at least:

- Suitable Product breakdown is available: A Product breakdown provides a structured representation of the Product. Refer to Chap 4.

- Description of Product use: An Operational Requirements Document (ORD) and/or a Customer Requirements Document (CRD) describe in detail the Product use. These documents provide information about project-specific operational and maintenance environment. Refer to Chap 3.

Note

If an ORD and/or CRD is not available for a project, it is necessary to document the description of Product use in an alternative way.

- Statistical data on special events: Data can be based, for example, on feedback from current or previous similar projects that quantify the occurrences of special events.
- Inputs from Preventive Maintenance Analysis (PMA): Applicable and effective PMTRE (refer to S4000P) and special event impact in the context of potential damage caused by a special event.

2.4 Special event analysis process - outputs

A tabular report is one of the possible outputs of the special event analysis. Such report provides all data/information developed during the special event analysis, in a structured and readable format. In order to keep the traceability of the identified maintenance task requirements and their baselining analyses, it is recommended to record these results as part of the LSA data.

Fig 2 provides a simple example of a tabular report approach, which documents the results of a special event analysis process. Table 5 contains an explanation of the content of each column.

Special event description	Propability of special event occurrence	Product item (LSA candidate) impacted	Special event impacts or damage description	Preventive Maintenance Task Requirement Event (PMTRE)	Corrective maintenance task requirement	Acceptable damage size description
1	2	3	4	5	6	7
Lightning strike	Remote likelyhood	Equipment AA	Damage of equipment AA	PMTRE-01 Inspection AA1 required	Damage repair for equipment AA	N/A
		Structural element S1	Damage of structural element S1	PMTRE-02 Inspection S1A required	Standard repair procedure XY for structural element type SE-003.	N/A

ICN-B6865-S3000L0069-003-01

Fig 2 Example on a special event analysis tabular report

It is recommended to adapt the tabular report as shown in Fig 2 to meet project specific requirements.

Note

Although not shown in Fig 2, it is recommended to include additional information in the report, for example release date, issue number, author's name, approval signatures and information required for the identification of the maintenance task.

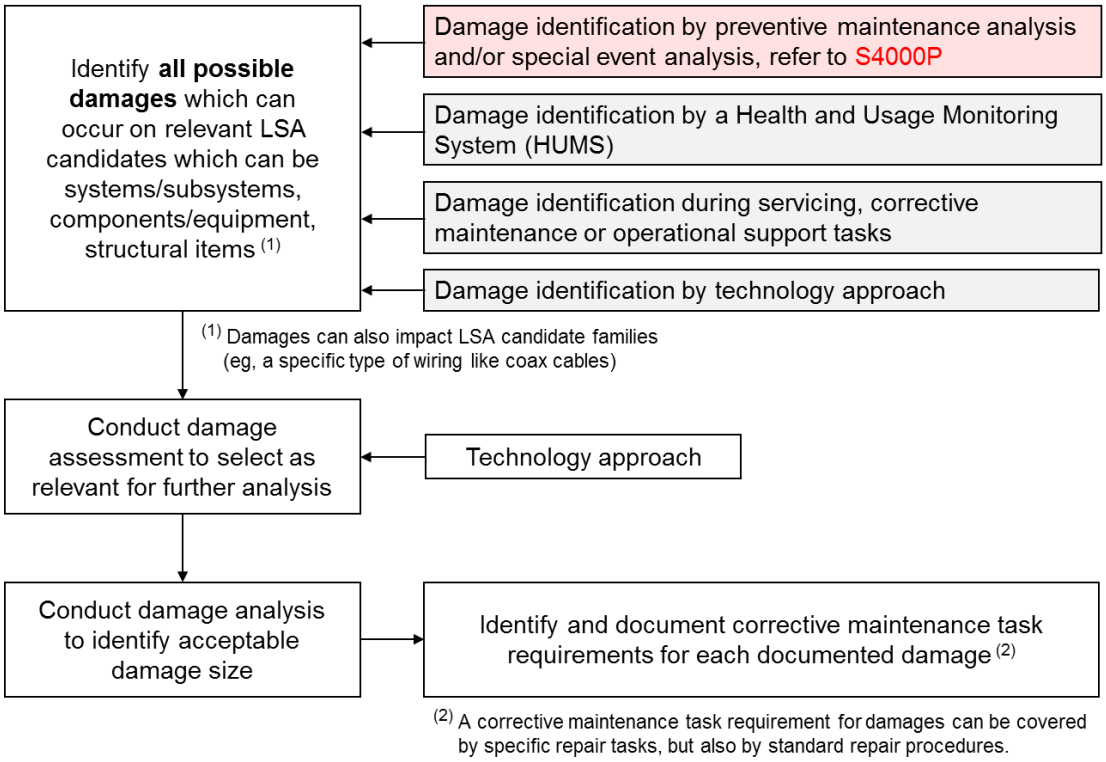
Table 5 Explanation on the tabular report for a special event analysis

Column	Description
1	**Special event description** The special event is described by a complete name. The special event can affect more than one element of the breakdown structure. In this case, identify each element on a different line.
2	**Probability of special event occurrence** The rating and description of the occurrence.
3	**Product item (LSA candidates) impacted** Element name. The breakdown element identifier can follow the element name. It is possible to group some elements (typically structural elements) into families in order to minimize the number of entries in the table.
4	**Special event impacts or damage description** A short description of impacts and potential damage. If there are multiple impacts/damage that can affect the same element, identify each impact/damage on a different line.
5	**Preventive Maintenance Task Requirement Event (PMTRE)** PMTRE includes all task requirements to consider after a special event has occurred, in order to check if any element needs to be repaired/restored before re-entry into service.
6	**Corrective maintenance task requirement** A short summary or title including the type of maintenance task requirement and the element involved. If more than one maintenance task is needed to cover the same special event, identify each task on a different line. Conversely, the same maintenance task can address more than one special event. This is typically the case of a standard repair procedure.
7	**Acceptable level of damage description** A qualitative or quantitative description of damage that can remain without repair, and damage that can be repaired.

3 Damage analysis

3.1 Damage analysis logic

The damage analysis process identifies relevant damage and the corresponding corrective maintenance task requirements to repair a detected damage. Refer to Fig 3.



ICN-B6865-S3000L0146-001-01

Fig 3 General flow chart for damage analysis logic

3.2 Damage analysis process

The damage analysis process involves the following activities:

- damage identification, refer to Para 3.2.1
- damage analysis to identify the allowable level of damage, refer to Para 3.2.2
- damage analysis to identify the corrective maintenance task requirements, refer to Para 3.2.3

3.2.1 Identification of relevant damage

The ways to identify the relevant damage to be considered include, but are not limited to:

- PMA activities in the context of Preventive Maintenance Task Requirements (PMTR) definition, refer to S4000P
- The technology approach, refer to Para 3.2.1.3

3.2.1.1 Identification of relevant damage by preventive maintenance analysis

PMA activities include analysis processes that identify the Preventive Maintenance Task Requirements Interval (PMTRI). Refer to S4000P:

- system analysis
- structure analysis
- zonal analysis

Each analysis process listed above can identify relevant damage on Product items as potential failure causes. This damage requires preventive maintenance in order to avoid functional failures that can lead to critical situations regarding operational safety, mission accomplishment, environmental integrity, or economic factors. This potential damage also requires corrective maintenance and information regarding the acceptable level of damage.

Within PMA activities, there is an additional analysis process to identify PMTRE, refer to S4000P.

3.2.1.2 Special event analysis

The special event analysis process can identify relevant damage on Product items as a potential consequence caused by a special event. This damage requires preventive maintenance to avoid the risk of using the Product in an unsafe or unclear condition after a special event has occurred. This potential damage also requires corrective maintenance and information regarding the acceptable level of damage.

Note

Para 3.2.1.3 describes a further analysis method, which uses a technology approach. It is also possible to use this method to evaluate the relevant damage modes identified by PMA and requiring further analysis. This is helpful to limit analysis effort according to the project requirements.

3.2.1.3 Identification or assessment of relevant damage by technology approach

The process described in this paragraph can serve two different purposes:

- the identification of the damage to consider for the identification of the corrective maintenance task requirements (eg, if results of the PMA are not available for damage identification)
- the evaluation of previously identified damage by the PMA to determine which damage is relevant for further analysis, with a view to identifying the corrective maintenance task requirements and the acceptable level of damage

The analysis process also evaluates the behavior of the technologies used in the Product design. Different technologies can show significantly different behaviors with respect to the impact of damage. The analysis approach identifies the technologies and analyzes them thoroughly from two points of view:

- Is the technology well known or new?
- Is the technology tolerant or sensitive concerning damage?

Identify the potential damage to be considered using the following five step methodology:

Step 1: For all Product items (systems, subsystems, structural parts, equipment, components), identify the technologies used and indicate the level of knowledge regarding their behavior.

Table 6 provides a qualitative approach to characterize this knowledge. The level of accuracy of the rating can require some adjustments based on the project requirements.

Table 6 Example of technology behavior rating

Rating	Technology behavior knowledge	Description
1	Well known	This technology has been used on many similar projects
2	Known	This technology has been used on similar projects, but it has been slightly modified for the intended project
3	New	This technology has been already used on similar recent projects, but little feedback is available, or the technology is completely new

Step 2: Evaluate the sensitivity of the technology.

For each technology identified in Step 1, identify the possible damage types that can occur. For example:

- general surface damage (eg, scratches, dents or cracks)
- breakage of structural components or controls
- tear off of connectors of electrical or data wiring
- corrosion on metallic parts (general, galvanic or other types of corrosion)
- stress corrosion on a metallic assembly with constraints installed
- delaminating or humidity absorption on composite material

Evaluate its sensitivity to damage types in relation with possible damage sources.

Table 7 provides a qualitative approach to characterize this sensitivity. The level of accuracy of the rating can require some adjustments based on the project requirements.

Table 7 Example of damage sensitivity rating

Rating	Sensitivity degree	Description
1	Extremely low	This technology has an extremely low chance of damage during Product life
2	Low	This technology has a low chance of damage during Product life
3	Medium	This technology has a medium chance of damage during Product life
4	High	This technology is likely to be damaged during Product life
5	Extremely high	This technology is very likely to be damaged during Product life

Step 3: Select the association technology behavior/damage sensitivity relevant for further analysis.

It is possible to identify a selection threshold by combining the technology behavior knowledge rating and the technology sensitivity rating (refer to Fig 4). The selection threshold is adjustable depending on the needs of the individual project.

		Sensitivity				
		1	**2**	**3**	**4**	**5**
Technology behavior	**1**	1	1	2	3	4
	2	1	2	3	3	4
	3	2	3	4	4	4

ICN-B6865-S3000L0088-002-01

Fig 4 Example of a technology/damage evaluation rating

Step 4: For all items of the Product breakdown identified in the previous steps, analyze their installation design and installation area in order to evaluate their exposure to the different threats.

For example, the level of exposure to corrosion depends on the location of the structural part. Items in a landing gear bay are highly subject to corrosion, whereas to a significantly lesser extent in watertight bays as they are opened rarely.

Step 5: Criticality analysis

In order to focus on the most significant damages, it is recommended that a criticality analysis be considered. For each potentially damageable item, it is recommended to evaluate the criticality in terms of impact on the Product operation using the following questions:

- Does the damage result in lowering safety level?
- Does the damage result in lowering availability?
- Does the damage result in faster deterioration?

Note

The list of questions can change to adapt to specific project needs.

Step 6: Using the results of the previous steps, decide which potentially damageable item is relevant for further analysis.

3.2.2 Identification of allowable level of damage

The previous analysis steps identified and selected the Product breakdown items that can be affected by damage during service life. These candidates can be either single items or families of items, for example structural items, wiring or tubing in a circuit.

It is possible to analyze each selected item to determine the acceptable level of damage:

- Which damage level does not require any corrective maintenance or can be left as it is, either with or without a cosmetic treatment?
- Which damage level can/must be repaired immediately by specific or standard repair procedures?
- Which damage level leads to a final discard of the item?

Typically, design department experts perform this evaluation, which can be either qualitative or quantitative. It can be based on measurements, calculations, tests or best engineering judgment.

3.2.3 Identification of corrective maintenance task requirements

At this point, it is necessary to analyze each significant item in order to determine the corresponding corrective maintenance task requirements.

This analysis identifies the corrective maintenance tasks required to:

- detect and measure exactly each damage that can occur (eg, by visual inspection, borescope inspection, non-destructive material testing like X-ray or eddy current inspection)
- restore the function of the item under analysis (eg, by standard repair task, refurbish task or item replacement, if necessary)

Note

In addition to specific damage detection procedures, damage can also be detected during normal Product operation, for example during routine inspections or corrective maintenance tasks. It is also possible to implement a Health and Usage Monitoring System (HUMS). For this purpose, it is necessary to install sensors or to carry out exact measurements that can warn the operator on potential damage more effectively.

Note

It is possible to group some items into families, and the associated damage is a candidate for standard repair procedures (eg, standard repair procedures for structural parts, standard repair procedures for wiring).

3.3 Damage analysis process - inputs

Damage analysis requires the following inputs:

- Product definition: Suitable Product breakdown available, including information about technologies used within the different Product elements (eg, systems, subsystems, structural items, equipment and/or components).
- inputs from PMA: Potential damages which can be identified as failure causes for functional failures or potential damage as a consequence of special events

3.4 Damage analysis process - outputs

A tabular report is one of the possible outputs of the damage analysis. Such report provides all the data/information developed during the damage analysis process, in a structured and readable format. In order to keep the traceability of the identified maintenance task requirements and their baselining analyses, it is recommended that these results be recorded, as part of the LSA data.

Fig 5 provides an example of a damage analysis tabular report. Table 8 provides a description of each column.

Product item (system/subsystem, structural item, equipment, component)	Damage description	Damage assessment description	Allowable level of damage	Corrective maintenance task requirement
1	2	3	4	5
Car fender	Minor scratch	• Damage not safety critical • Damage not operational critical • Cosmetic repair required	Scratch just on the surface, metallic structure not visible, basic painting not impacted	Cosmetic standard repair task (painting)
	Heavy scratch	• Damage not safety critical • Damage not operational critical • Damage must be repaired within 2 weeks to prevent corrosion impact	Scratch goes down to the metallic structure, painting completely removed, risk of corrosion given	Verification of scratch depth Standard repair task to correct deep scratches

ICN-B6865-S3000L0147-001-01

Fig 5 Example of damage analysis tabular report

The tabular report above needs some adjustment to meet the project specific requirements.

Note

Although not shown in Fig 5, it is recommended that additional information be included in the report, for example release date, issue number, author's name, approval signatures and information required for the identification of the maintenance task.

Table 8 Explanations of the example of damage analysis tabular report

Column	Description
1	**Product item (system, subsystem, structural item, equipment, component)** Element name. This element name can be followed by its breakdown element identifier. It is possible to group some elements (typically structural elements) into families in order to minimize the number of entries in the table.

Column	Description
2	**Damage description** A short description of damage. If different types of damage can affect the same element, identify each type on a different line.
3	**Damage assessment description** A list of qualitative or quantitative elements that supports the decision to select or discard a case.
4	**Allowable level of damage** There can be more than one value. The value of damage that can be left as it is, and value of damage that can undergo maintenance.
5	**Corrective maintenance task requirements** A short summary or title including the type of maintenance task and the element to which the maintenance task is applied. If more than one maintenance task is needed for the same item, identify each task on a different line. Conversely, the same maintenance task can address more than one item. This is typically the case of a standard repair procedure.

4 Influence on design

Damage and the impact of special events can have significant influence on Product operation and availability. Normally, it is not possible to prevent damage and special events from occurring at all. However, the reduction/limitation of the impact of damage and special events must be a common goal of design and development and supportability engineering. For this purpose, there is a need to consider the potential impact of damage and special events already during the early phases of Product design. Examples to improve the tolerance against damage and special events include, but are not limited to:

- if the damage tolerance is low and the Product is operated under rough conditions, consider the use of highly sophisticated technology. Consider using different technology if the risk of damage is too high for sensitive technology used under rough conditions.
- enhance protection for outside structural items which have a high probability of being affected by damage
- design protections for sensitive equipment (eg, covers, appropriate mounting, padding)
- improve storage and transport conditions of the Product or Product components by a corresponding robust design or by appropriate means of transport
- improve accessibility to ease visual or other inspections in case you suspect damage has occurred
- prefer the use of standardized components with well-known technology which can easily be repaired by simple standard repair procedures

5 Relevant elements of the S3000L data model

The following Units of Functionality (UoF) support the documentation of the data associated to this chapter. Refer to Chap 19:

- S3000L UoF Damage Definition
- S3000L UoF Environment Definition
- S3000L UoF Product Usage Context
- S3000L UoF Product Usage Phase
- S3000L UoF Special Event
- S3000L UoF Task Requirement
- S3000L UoF Task Usage
- S3000L UoF Time Limit

Chapter 9

Operational support analysis

Table of contents

List of tables

List of figures

References

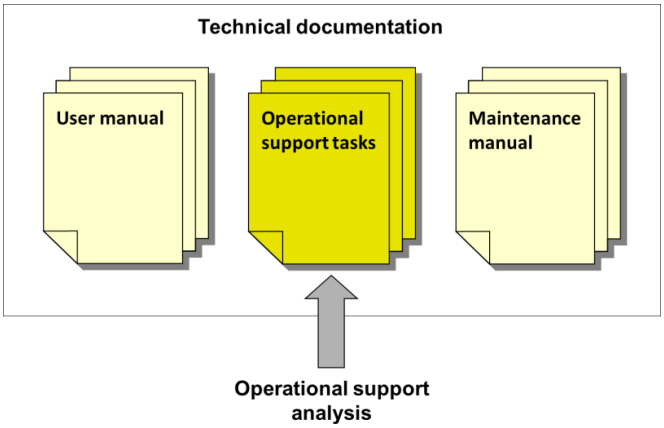
Table 1 References

Chap No./Document No.	Title
Chap 3	LSA process
Chap 10	Development of a preventive maintenance program
Chap 12	Task requirements and maintenance task analysis
Chap 13	Software support analysis
Chap 16	Disposal
Chap 19	Data model
Chap 21	Terms, abbreviations and acronyms
S1000D	International specification for technical publications using a common source database

1 General

1.1 Introduction

With respect to Product operation and handling, there are additional aspects to be considered beside Product maintenance and repair activities. Operational support tasks include activities performed in areas other than the area where the Product is used (documented in operating instructions) or maintained (documented in a maintenance manual). However, these tasks can be vital for the proper usage of any Product. Operational support tasks influence many aspects, including ease of operation, usability, flexibility of usage, or mobility. There is a thin line between Product use and operational support tasks. This line changes on a project basis and it is necessary to document these aspects. Sometimes it is difficult to determine whether a task relates to use or operational support.



ICN-B6865-S3000L0039-002-01

Fig 1 Operational support tasks within technical publications

All tasks identified by the operational support analysis will fill a possible gap between user manual and maintenance manual. Refer to Fig 1. In any project, it is recommended to clarify at an early stage who will be responsible for the supportability analysis and documentation for operational support tasks.

1.2 Purpose

This chapter is a guideline for identifying relevant operational support activities. Its target readers are support personnel responsible for analyzing the relevant operational support tasks. Para 4 includes a detailed list of potential activities to support analysts.

1.3 Scope

This chapter provides details and examples on the most common operational support tasks. There can be more activities, especially in relation to environmental conditions (eg, preparation for transportation under extreme climatic conditions such as an arctic environment). Each example includes a short description and underlines specific aspects to be considered.

1.4 Terms, abbreviations and acronyms

Definitions that are specific to operational support analysis are given in Table 2. Chap 21 provides the complete set of LSA terms, abbreviations, and acronyms.

Table 2 Terms, abbreviations and acronyms

Term	Definition
Operational support task	In Product operation, any support activity that fulfils task requirements, such as servicing or Packaging, Handling, Storage and Transportation (PHST) requirements

2 Operational support tasks

An important aspect of supportability analysis activities is the identification of operational support tasks, including the requirements for personnel, support equipment, consumables, spare parts, facilities and required training. Some tasks must be considered early in the design and development process. Some tasks can be considered later, for example when a prototype for the Item Under Analysis (IUA) is available.

2.1 Activities in supporting Product use

2.1.1 Preparation for usage

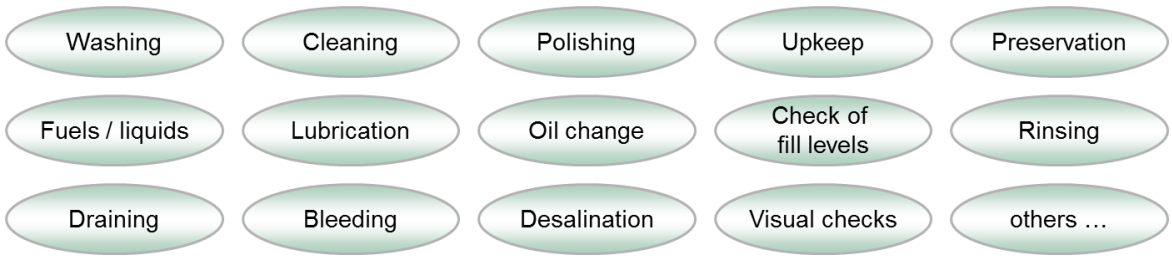
The tasks required to prepare a Product for usage/operation can include, for example, the exchange of equipment to provide a special capability (also referred to as role change). Product modification can occur by changing some pieces of equipment. Additionally, peripheral Products/components must be prepared for proper usage.

Typical preparation tasks include servicing tasks, like cleaning a windshield before driving. Other typical preparation tasks are the complete conversion of a Product for another use as the change of the tooling within a machine to produce a new product line. Typical examples of tasks concerning preparation for use and included in operational support analysis are:

- preparation of machines for a new production segment (eg, change in tooling and dies)
- change of vehicle category (eg, a passenger vehicle turned into an ambulance)
- preparation of an aircraft for a reconnaissance mission by means of special equipment
- preparation of a ship including the required equipment for sailing

2.1.2 Servicing

The term servicing can describe a wide range of tasks performed on a Product, also in connection with Product preparation for use. Refer to Fig 2. Other servicing activities include handling Products after usage or maintaining them. This can include, for example, a simple visual inspection for foreign objects during washing or cleaning procedures, as well as simple visual inspections to detect any damage or assess the equipment general conditions (eg, condition of tires).



ICN-B6865-S3000L0040-002-01

Fig 2 Typical servicing tasks

Maintenance Task Analysis (MTA) must include servicing tasks, if existing. Refer to Chap 12. Servicing tasks can have an important impact on resources such as personnel, support equipment (eg, for lubrication), consumables and facilities (eg, washing facilities with oil/water separator and water recycling equipment). Typical examples for servicing tasks are:

- washing an engine
- gear lubrication
- changing engine oil
- product polishing and subsequent preservation
- brake bleeding
- changing hydraulic liquid
- replenishing fluids
- desalination and equipment rinsing after diving

2.1.3 Adjusting

This type of task can also be performed in connection with Product preparation for usage. There are no changes in Product functionality, but precision and quality are evaluated as preconditions for Product usage. Typical examples of adjustment tasks are:

- leveling of the entire Product as a basic task

- calibration of measuring instruments
- adjustment of gun sight

2.1.4 Weighing

The analysis of weighing tasks involves the preparation of the IUA for weighing and the relevant weighing procedure. This includes information on the weighing equipment. The contractor and the customer must define and agree on the purpose of the weighing procedure (eg, to prepare a mission or prove compliance to Product specification requirements).

2.1.5 Loading and unloading

It is necessary to analyze loading and unloading procedures for each Product that can be used for cargo transportation. This information must be collected at an early stage of the life cycle development. It is necessary to provide examples of loading and offloading techniques, interior layout, floor loadings, location and strength of lashing points, stowing and securing methods, capacities and dimensions of compartment and doors. Analysis questions include, but are not limited to:

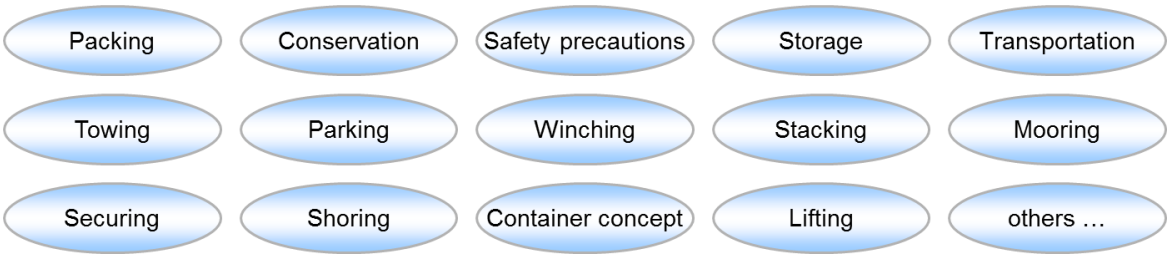
- What kind of cargo will be transported?
- Which size and weight parameters must be considered?
- Is the expected cargo sensitive to special impacts (acceleration, magnetic or electric fields, pushes, humidity, etc)?
- Is the cargo critical or even dangerous in case of improper handling?
- What type of cargo securing (eg, lashing and lashing points, stowing) is required?
- Will a cargo container concept be employed?
- Does the type of load require special loading devices (eg, rolling devices)?

It is also necessary to address the requirements for special support equipment for loading and unloading.

- Do loading/unloading operations require special support equipment or vehicles?
- Is special support equipment required to secure the cargo?

2.2 Packaging, handling, storage and transportation aspects

Fig 3 shows an overview of PHST activities/aspects which must be considered for both, the Product and Product components like equipment or assemblies.



ICN-B6865-S3000L0041-002-01

Fig 3 Typical PHST tasks

2.2.1 Packaging and unpacking

A packaging concept for IUAs and/or components includes, but is not limited to:

- Is packaging required for short-term storage and/or long-term storage?
- Is packaging required for transportation and what is the type of transportation?
- Is a special container required for storage or transport?
- Is special preservation required because of extreme climatic conditions during storage or transportation (eg, sea transport)?

- Is it required to unpack and repack the IUA during transportation or storage, for example, to perform maintenance activities (eg, storage tasks)?
- What is required for the unpacking and removal of preservation concerning support equipment and facilities?

2.2.2 Handling

Handling tasks for the IUA can include but are not limited to:

- safety precautions and limitations due to handling
- instructions to park, tow, winch or move the IUA for uncommon usage
- instructions to jack the IUA, including jacking points, required adapters, supports for special components, balance weights, jacking procedures
- additional equipment and materials required when handling the IUA (eg, tow bars or cables)

2.2.3 Storage

It is necessary to analyze and document storage procedures to guarantee product functionality during and after storage. The analysis will consider the following aspects:

- Is the required storage long term or short term?
- Is the Product/component to be stored sensitive?
- Is it necessary to remove components from the Product to be stored and to store these components separately under special conditions?
- What type of inspection and preventive maintenance are required to safeguard structural and Product integrity during storage (eg, wheel rotation, power source, engine running or pressure checks)? Provide a timescale for maintenance during storage.
- Are extreme climatic conditions expected during storage (heat, frost, humidity)?
- Are there any special techniques required before storing the Product (eg, cleaning and preservation, fluid system draining/replenishing, static grounding, protective blanking, removal of special components)?
- Are there any special techniques required to retrieve the Product from storage and bring it into operation (eg, cleaning and removing preservation devices, fluid system replenishing, re-installation of special components, functional checks, preparation for usage)?
- What kind of securing is required during storage (eg, mooring, blocking of movable components, and protection from light)?
- Is it necessary to consider the storage time in connection with the life of the stored Product/component?

2.2.4 Stacking

Stacking is a special aspect of storage and transportation of any Product/component. It is essential to address stacking safety requirements for storage or transportation. For example, the impact of external events such as pushing must be analyzed for storage.

2.2.5 Lifting

Lifting operations must include all necessary procedures to lift the IUA with relevant hoist devices, cranes, jacks or slings in case of:

- lifting for transportation or loading purposes
- lifting for repair or maintenance purposes
- lifting for recovery

In addition to the complete Product, it is necessary to consider the lifting of components as well (eg, lifting an aircraft engine).

2.2.6 Transportation

Based on customer requirements, the analysis of Product transportability must include:

- How much work is necessary and how long does it take to prepare the Product for transportation and recovery after transportation?
- What are the requirements for preparation for transportation and for recovery after transportation concerning personnel, tools, consumables, and facilities?
- What are the additional environmental requirements for transportation under special conditions (eg, extreme climatic conditions, including desert environments, humidity, heat, frost and transportation by sea)?
- What are the additional safety requirements for transport under special conditions (eg, loading securing within a transport aircraft)?
- What are the required servicing activities during transportation (especially during long journeys)?
- Is it necessary to remove components for transportation or to disassemble the entire IUA down to a specific level?
- Should a container concept be considered?
- Can sledges and/or pallets be used?

2.2.7 Mooring

Analysis of mooring tasks for the IUA must consider all weather conditions and any means of transportation. Mooring can be either long-term or short-term. The purpose of this activity is to tie down or secure by other means the IUA to the ground or within a vehicle to avoid any damage. The analysis must also include information on special techniques applicable to ballasting, definition of lashing points and installation/use of special support equipment applicable to mooring.

2.2.8 Shoring

Shoring analysis must include shoring points, procedures and equipment used during maintenance, repair and recovery.

2.3 Role change

A Product can be used in different contexts (eg, a transport/cargo helicopter). In this case, the operational support tasks focus on the tasks aimed at changing the configuration of the Product to facilitate its use in a specific context (eg, removal of seats, removal of intercommunications apparel and installation of specific equipment and corresponding fixing parts).

2.4 Deployment

Product deployment may need to take into consideration several tasks depending on the complexity of the Product itself. It can also include a system of systems that contains several Products. Deployment can relate to different activities, for example installation, transportation to the operational area, as well as preparation, operation setup and test of functionalities before operation. Therefore, the task analysis will vary for each Product based on aspects such as:

- operational scenario (eg, facility preparation, installation or preparation to operation, personnel competence for Product deployment)
- transportation capability (eg, fixing procedures tasks, safety procedures)
- preparation to operation (eg, remote or local setup tasks, calibrating procedures, tuning procedures)

2.5 Software and data aspects

Chap 13 addresses operations related to software support.

2.6 Product recovery

Recovery analysis must include any information on planned recovery procedures and the support equipment required to recover any IUA from any condition that can affect it.

Product recovery during the testing phase must be addressed separately. It is important to guarantee a fast reaction to any unexpected event that involves recovering the Product from an

undesired condition. A recovery plan for all emergencies must be detailed to avoid endangering personnel or the environment.

Recovery also includes rescue procedures or the safeguarding of an area after a catastrophic event. All procedures that are necessary to rescue any Product after an accident must be analyzed before the first use of the Product or a Product prototype. For example, an aircraft performing its first flight marks a specific risk of a crash. All required activities after a potential crash must be analyzed and documented carefully before the first use of the Product. The analysis process must include recovery training to reduce the risk of extended damage to personnel or the environment in case of a catastrophic event. All aspects must be covered, including accidents on land, in water (eg, foundering and the need to salvage the ship) or within areas that are difficult to access.

2.7 Special preventive maintenance

Chap 10 describes the handling of preventive maintenance in general terms. However, servicing or preparation for use tasks can be preventive in nature, yet Preventive Maintenance Analysis (PMA) does not identify them. Being performed frequently (eg, daily), these tasks can have a high impact, for example on personnel requirements. Typical examples of tasks are:

- Visual check before use (eg, engine start)
- Visual check after special missions
- Preventive replenishment of fuel, fluids or other consumables to guarantee proper function
- Cleaning before or after each use

2.8 Disposal and recycling

Analysis concerning requirements after the Product life cycle is becoming increasingly important. This includes both the disposal of the Product, and the handling of components and consumables to be disposed during the Product life cycle. Refer to Chap 16.

2.9 Extraordinary operational support tasks

The analysis process must address extraordinary operational support tasks. Typical examples are:

- Decontaminating a vehicle
- Disinfecting personnel before starting a job
- De-icing an aircraft or ship
- Checking electrical charge
- Checking the strength of magnetic fields during storage
- Completing paperwork for statistical purposes

3 Documenting operational support tasks

3.1 Data collection aspects

LSA data must document the identification of any operational support task requirement. To this end, it is possible to use appropriate existing information codes from S1000D. It is recommended that operational support tasks be documented at Product level or at the appropriate breakdown level. As a matter of fact, use of a mixed physical/functional breakdown provides the most flexibility. This supports the exchange of data with S1000D for technical publications.

3.2 Operational and customer requirements as a source

In general, the requirements for operational support tasks can be derived from the operational and customer requirements, which are collected and documented upon the gathering of Product use data. Refer to Chap 3. As a first input for operational support analysis, it is recommended to use the content of the relevant Operational Requirements Document (ORD) and Customer

Requirements Document (CRD). Each aspect covered in the ORD or CRD concerning use of the IUA can require an operational support task.

4 Checklist for operational support analysis

Table 3 provides an alphabetical checklist for potentially relevant activities.

Table 3 Checklist for support related operations analysis

Activity	Analysis recommended at which phase	Probable high impact on requirements for facilities	Probable high impact on requirements for special support equipment	Probable high impact on design
Adjusting	When necessary	No	Possible	No
Bleeding	When necessary	No	No	No
Calibration	When necessary	No	Yes	No
Checking	When necessary	No	No	Possible
Cleaning	Early phase	Yes	Possible	No
Conservation	Early phase	Yes	Possible	No
Container concept	When necessary	No	No	Possible
Conservation removal	Early phase	Yes	No	No
Desalination	When necessary	No	No	No
Disposal	Early phase	Possible	Yes	Yes
Draining	When necessary	No	Possible	Possible
Jacking	Early phase	Possible	Yes	Yes
Leveling	Early phase	Possible	Possible	No
Lifting	Early phase	Possible	Yes	Possible
Loading cargo	Early phase	Yes	Yes	Yes
Loading data	Early phase	Yes	Possible	No
Loading Software	When necessary	No	Possible	No
Lubrication	When necessary	No	Possible	Possible
Mooring	When necessary	No	Yes	Yes
Oil Change	When necessary	No	No	No
Packing	When necessary	Possible	Possible	No
Parking	When necessary	Possible	Possible	No
Polishing	When necessary	No	Possible	No

Activity	Analysis recommended at which phase	Probable high impact on requirements for facilities	Probable high impact on requirements for special support equipment	Probable high impact on design
Preservation	Early phase	Possible	Possible	No
Recovery	Early phase	No	Possible	Possible
Recycling	When necessary	Possible	Possible	Yes
Replenishment	When necessary	No	Possible	Possible
Rinsing	When necessary	Possible	No	No
Safety precautions	Early phase	Possible	Possible	No
Securing	When necessary	No	Possible	Possible
Shoring	When necessary	No	Possible	Possible
Stacking	When necessary	No	Possible	No
Storage	Early phase	Yes	Possible	No
Towing	When necessary	No	Yes	Possible
Transportation	Early phase	Possible	Yes	Possible
Unloading cargo	Early phase	Possible	Yes	Yes
Unloading data	When necessary	No	Possible	No
Unloading Software	When necessary	No	Possible	No
Unpacking	Early phase	Possible	Yes	No
Upkeep	Early phase	Possible	Possible	No
Washing	Early phase	Possible	Possible	No
Weighing	Early phase	Possible	Yes	No
Winching	When necessary	No	Possible	No

5 Associated parts of the S3000L data model

The documentation of the associated data to this chapter is supported by the following Units of Functionality (UoF), refer to Chap 19:

- S3000L UoF Environment Definition
- S3000L UoF LSA Candidate
- S3000L UoF Product Usage Context
- S3000L UoF Task Requirement
- S3000L UoF Task Usage
- S3000L UoF Time Limit

Chapter 10

Development of a preventive maintenance program

Table of contents

List of tables

List of figures

References

Table 1 References

Chap No./Document No.	Title
Chap 8	Special event and damage analysis
Chap 11	Level of repair analysis
Chap 12	Maintenance task analysis
Chap 14	Life cycle cost considerations
Chap 19	Data model
S1000D	International specification for technical publications using a common source database
S4000P	International specification for developing and continuously improving preventive maintenance
DEF-STAN 00-45 Part 3	Using Reliability Centered Maintenance to Manage Engineering Failures, Part 3, Guidance on the Application of Reliability Centered Maintenance
MIL-STD 1843	Reliability-centered maintenance for aircraft, engines and equipment
MIL-STD 2173	Reliability-centered maintenance for naval aircraft, weapon systems and support equipment

1 General

1.1 Introduction

Product preventive maintenance comprises preventive maintenance at regular intervals and maintenance due to unexpected special events. Preventive maintenance tasks initiated by an interval (often referred to as scheduled maintenance) aim to achieve continued Product safety, compliance with the law, protection of the environment, operational/mission availability, and cost-effective feasibility during a Product's in-service phase.

A PMTRI result from the application of analysis methodologies as described in other specifications, for example in S4000P. PMTRI are allocated to relevant and effective scheduled

task packages in order to implement an effective Preventive Maintenance Program (PMP) and optimize the total preventive maintenance effort for a Product. If required, packaging rules and solutions must comply with requirements from regulatory authorities.

S4000P specification provides an analysis methodology for special events, which generates PMTRE. All relevant special events are identified during the LSA process. Refer to Chap 8. After each special event, it is necessary to apply an analysis logic to identify an applicable and effective set of PMTRE. This will ensure safety, compliance with the law, environmental integrity, operational/mission availability and best economic feasibility for further Product use after a special event. PMTRE are not included in the packaging process for PMTRI.

1.2 Purpose

The purpose of this chapter is to define the process, and describe basic rules for the packaging of PMTRI to implement an effective PMP for the Product.

Note

Different projects, customers and companies use different terminology for PMP. Alternative terms are Preventive Maintenance Plan (PMP), Operator Maintenance Program or Operator Maintenance Plan (OMP), Scheduled Maintenance Program (SMP), all of them having the same meaning.

1.3 Scope

This chapter describes the process and basic rules to adapt preventive maintenance intervals and develop scheduled task packages based on all identified PMTRI. It is essential to ensure traceability from each PMTRI to the resulting scheduled maintenance task within a scheduled task package (and audit trail). Operator's input and decisions determine the need to adapt intervals for single PMTRI. Typically, customers/operators specify an interval target value for each scheduled task package based on availability and/or economic requirements.

In addition to the above-mentioned packaging process, it is necessary to perform Maintenance Task Analysis (MTA) on all scheduled maintenance task packages to get an estimate of the Product's scheduled maintenance effort.

2 Development of PMTRI

2.1 General analysis processes

Existing specifications concerning the identification of preventive maintenance requirements (such as S4000P, DEF-STAN 00-45 Part 3, ATA/A4A MSG-3, MIL-STD 1843, and MIL-STD 2173) contain three analysis processes covering the complete Product under analysis, along with Product development activities:

- system analysis
- structure analysis
- zonal analysis

2.1.1 System analysis

The system analysis aims at developing applicable and effective PMTRI for Product systems, subsystems and included equipment and components. The Product systems are analyzed in a structured and transparent approach to prevent failures, which can be critical to safety, compliance with the law, environmental integrity, operational/mission availability and high economic damage. The central goal of system analysis is to identify effective and applicable PMTRI to avoid such critical failures.

Refer to S4000P for extended information concerning the system analysis process.

2.1.2 Structure analysis

The structure analysis aims to develop applicable and effective PMTRI for the mechanical Product structure (eg, the body of land vehicles, airframes, or ship hulls). Based on the consequences of their loss of function, there are three types of Product structure:

- structural components relevant for Product safety - Structure Significant Items (SSI)
- structural components with impact on Product maintenance
- structural components with no significant impact on Product operation

Refer to S4000P for further information concerning the structure analysis process.

2.1.3 Zonal analysis

The zonal analysis considers additional aspects not covered by system analysis or structure analysis to identify further PMTRI assigned to a zone. It takes into account standard impact parameters for a zone of a Product and gives appropriate attention to additional aspects as required. As laid out in S4000P specification, the aspects to be considered include, but are not limited to:

- susceptibility to damage for hardware located in a specific zone
- density of equipment within a zone
- sources of ignition in a zone (eg, caused by damages on electrical wiring)
- combustible material and/or vapor accumulation as a risk factor in a zone
- impact of lightning or High Intensity Radiated Fields (HIRF) within a zone

Refer to S4000P for further information concerning the zonal analysis process.

2.1.4 Additional sources for PMTRI

In addition to the identification of PMTRI based on analysis methodologies (included, for example, in S4000P), there are further sources to justify a PMTRI, such as:

- Certification Maintenance Requirements (CMR) including (national/international) law provisions and requirements from regulatory authorities
- requirements from Product safety analysis/Fault Tree Analysis (FTA)
- structural inspections related to material fatigue
- Original Equipment Manufacturer (OEM) recommendation/requirements (often a prerequisite for warranty)
- best engineering judgment
- experience from other projects
- PMTRI selected by individual users
- recommended scheduled activities during storage periods

2.2 Time limits, thresholds and triggers for a scheduled task

2.2.1 General

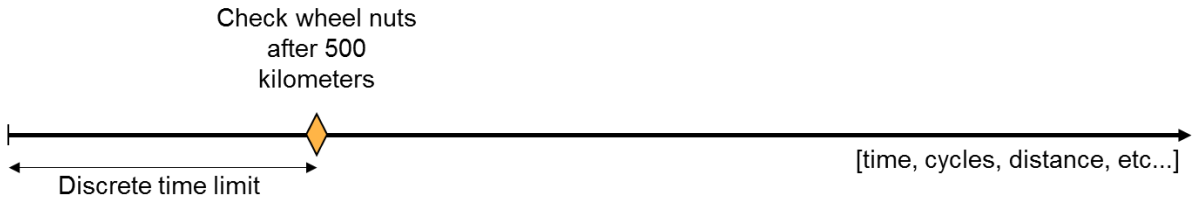
Scheduled tasks are determined for PMTRI identified by PMA. For each scheduled task, it is possible to assign different types of time limits, thresholds and triggers to:

- discrete time limits (eg, time limits which are performed only once)
- periodic time limits, divided in:
  - initial time limit (specific interval for the first scheduled performance of a periodic task)
  - repeat time limits (time limits repeated at specific intervals)
- Triggers and thresholds

Triggers and thresholds can be defined as parameter values (eg, 1000 operating hours) or based on an event (eg, after performing a specific task, after a special event has occurred a number of times).

2.2.2 Discrete time limit

A task satisfying a PMTRI that must be performed only once is characterized by a discrete time limit. For example, a preventive maintenance task to check wheel nuts after a certain driving distance or to check the torque of special screws after a certain time of Product usage. Refer to Fig 1.

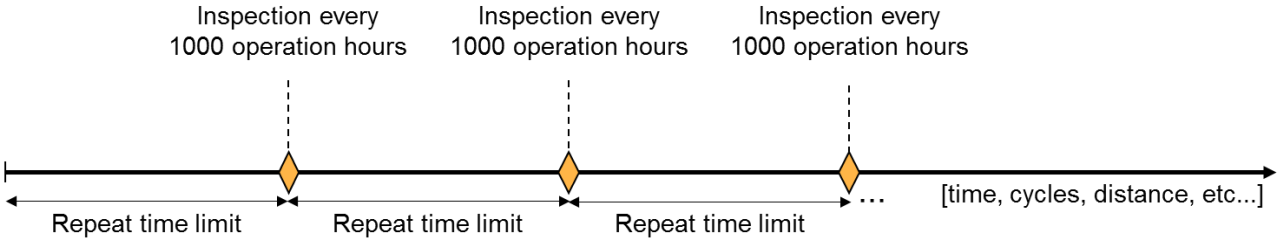


ICN-B6865-S3000L0043-003-01

Fig 1 Discrete time limit

2.2.3 Periodic time limit - repeat time limit

A task satisfying a PMTRI that must be performed repeatedly with a constant interval during the lifetime of a Product is characterized by a repeat time limit. Therefore, it will have a repeated threshold, known as a classical interval. Typical tasks are, for example, a periodic inspection or a periodic servicing task, such as an engine oil change. Refer to Fig 2.

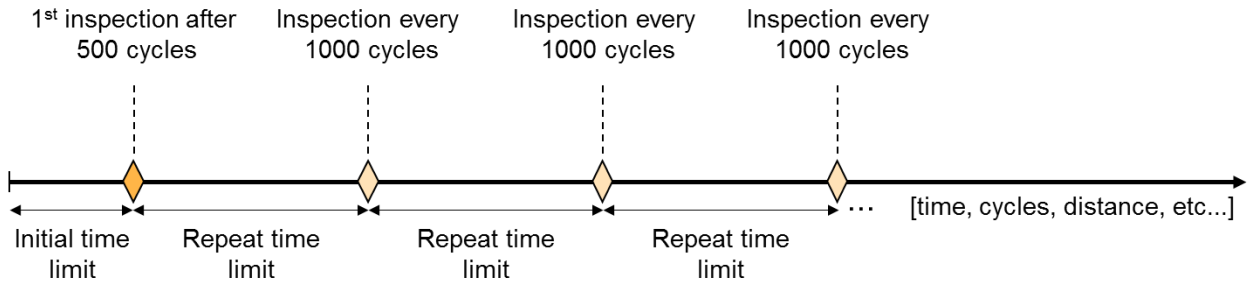


ICN-B6865-S3000L0044-003-01

Fig 2 Repeat time limit

2.2.4 Periodic time limit - initial and repeat time limit in combination

A combination of initial and repeat time limits is a common Product maintenance practice, for example, for Product structures. LSA data must document data for both time limits. Refer to Fig 3.



ICN-B6865-S3000L0045-003-01

Fig 3 A combination of tasks with initial and repeat time limits

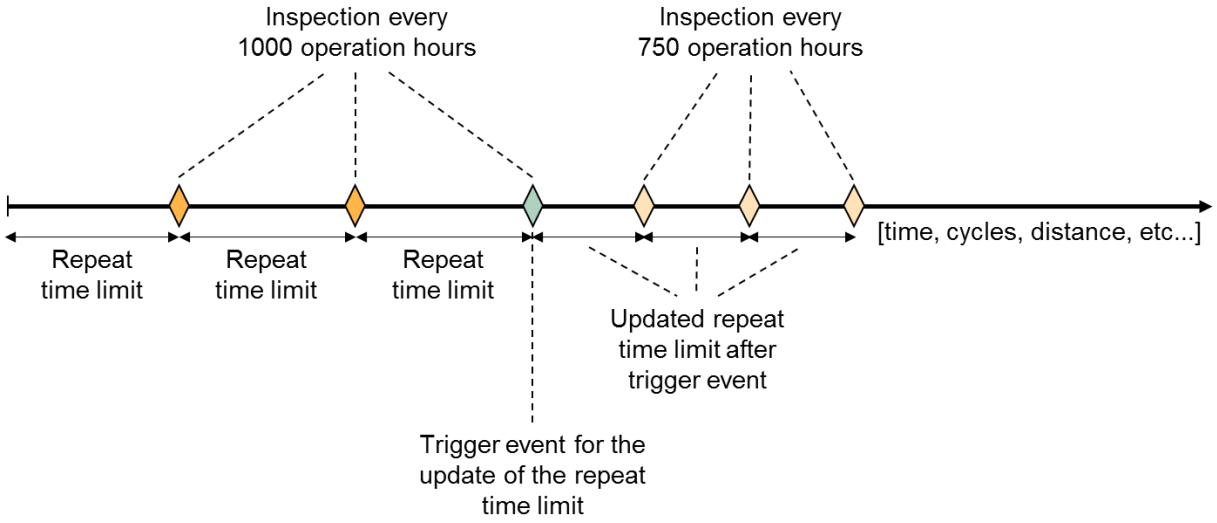
The start of the repeat time limit depends on the maturity of the initial time limit and on the experience gained from the efficiency of performing the corresponding preventive maintenance task for the first time.

Note

The examples from Fig 1 to Fig 3 show how to define a threshold using a numerical value and a corresponding unit of measure. A second specialization is to refer to the number of occurrences of an event to set a threshold (eg, occurrence of a special event, performance of a specific task).

2.2.5 Trigger events

Defining a trigger event can lead to the need to change time limits during the complete Product life cycle. A trigger event can be based on a numeric parameter or on the occurrences of an event (eg, a task, a special event, or a failure). The trigger can modify a time limit, but also change the type of scheduled maintenance task. Refer to Fig 4.



ICN-B6865-S3000L0046-003-01

Fig 4 Example of a trigger event

A typical use of a trigger event is the consideration of familiarization, learning phases and phases occurring after a certain period in the life of a Product, if for example, deterioration effects can force the implementation of modified PMTRI.

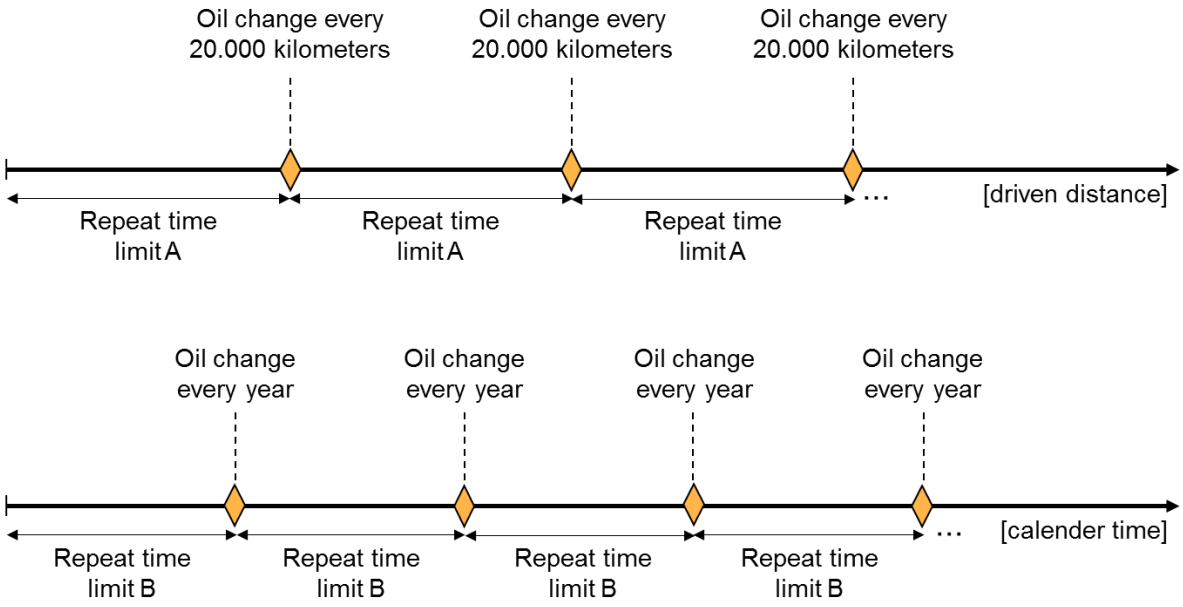
Note

Furthermore, a trigger event can terminate a PMTRI. This means there will not be any preventive follow-on task after the trigger event.

2.2.6 More than one periodic time limit

Depending on the deterioration process affecting equipment and/or components of a Product, a PMTRI can be limited to more than one threshold value. In several cases analysts define an interval based on usage parallel to a calendar-based interval. This approach is often referred to as a "whatever comes first" situation. In this case, the time limit that comes first during Product usage is the one that triggers the corresponding scheduled maintenance task.

Example:
A Product manufacturer determines the need to change the engine oil of a vehicle either after driving 20.000 kilometers (repeat time limit A), or after one calendar year (repeat time limit B). Both thresholds are a factor in engine oil deterioration, and have different impact parameters. For example, if the Product's driven distance is only 8000 kilometers per year, the calendar-based interval (1 year) launches the scheduled oil change. If the Product's driven distance is already 20.000 kilometers after only six months, the usage-based interval (20.000 kilometers) launches the scheduled oil change. Refer to Fig 5.



ICN-B6865-S3000L0111-002-01

Fig 5 More than one periodic time limit for a preventive maintenance task

3 PMTRI within the LSA

As described in Para 2, a PMTRI is developed on analytical basis (eg, based on S4000P). The analysis methodologies use selection logic to identify applicable and effective preventive maintenance task requirements. After a harmonization and consolidation phase (refer to S4000P), LSA data document the remaining PMTRI based on system, structure and zonal analysis (refer to Para 2.1.1, Para 2.1.2 and Para 2.1.3) and based on additional sources, refer to Para 2.1.4.

3.1 PMTRI from system analysis

PMTRI identified by a system analysis are allocated to the impacted Breakdown Element (BE) or part. The relevant system, subsystem, equipment or component represented by a BE or a part within the LSA data becomes an LSA candidate.

3.2 PMTRI from structure analysis

To document PMTRI for structural items within the LSA data, it can be necessary to include limited areas or Structural Details (SD) in the Product breakdown for the identified structural items. Documenting an SD within the Product breakdown requires the creation of additional BE representing structural areas. These additional LSA candidates are necessary to enable the correct assignment of the identified PMTRI for these structural areas. Refer to S4000P.

3.3 PMTRI from zonal analysis

PMTRI identified during zonal analysis can be allocated to:

- the 3-dimensional zonal areas of the Product under analysis (Product zones)
- selected equipment/items of Product systems
- selected structural items of a Product

Product zones can comprise equipment and components from different systems and structural items of a Product. It is possible that some selected Product zones only contain structural items.

It is recommended that a corresponding BE for each identified Product zone (eg, within an applicable Product zonal plan) be assigned. In general, each zone is a potential LSA candidate. As a matter of fact, the standard zonal analysis normally defines a General Visual Inspection

(GVI) with a scheduled interval for each Product zone. The manufacturer and/or responsible authorities must justify and agree to any exception.

PMTRI resulting from zonal analysis and allocated to selected equipment/items of Product systems or to structural items of a Product are handled in the same way as the PMTRI resulting from a system analysis or from a structure analysis.

3.4 LSA activities for a PMTRI

3.4.1 Criticality of the PMTRI

For each identified PMTRI, it is necessary to document the "worst-case" criticality of any Functional Failure Effect (FFE) that can occur at Product system level, in order to enable the packaging process described in Para 4 and ensure traceability at a later stage.

Note

It is possible to use a Functional Failure Effect Code (FFEC) assigned by S4000P analysis activities as an example for the documentation of PMTRI criticality.

3.4.2 Scheduled maintenance task and MTA

All PMTRI allocated to LSA candidates subsequently drive the corresponding rectifying scheduled maintenance tasks. In the LSA data, each PMTRI is linked to one or more scheduled maintenance tasks. All these tasks undergo an MTA. Refer to Chap 12.

In case an MTA for a single scheduled maintenance task detects potential problems concerning the applicability and effectiveness of the task, it is necessary to provide feedback to the analyst responsible for the PMTRI development to clarify the origin of the assumptions related to task applicability and effectiveness:

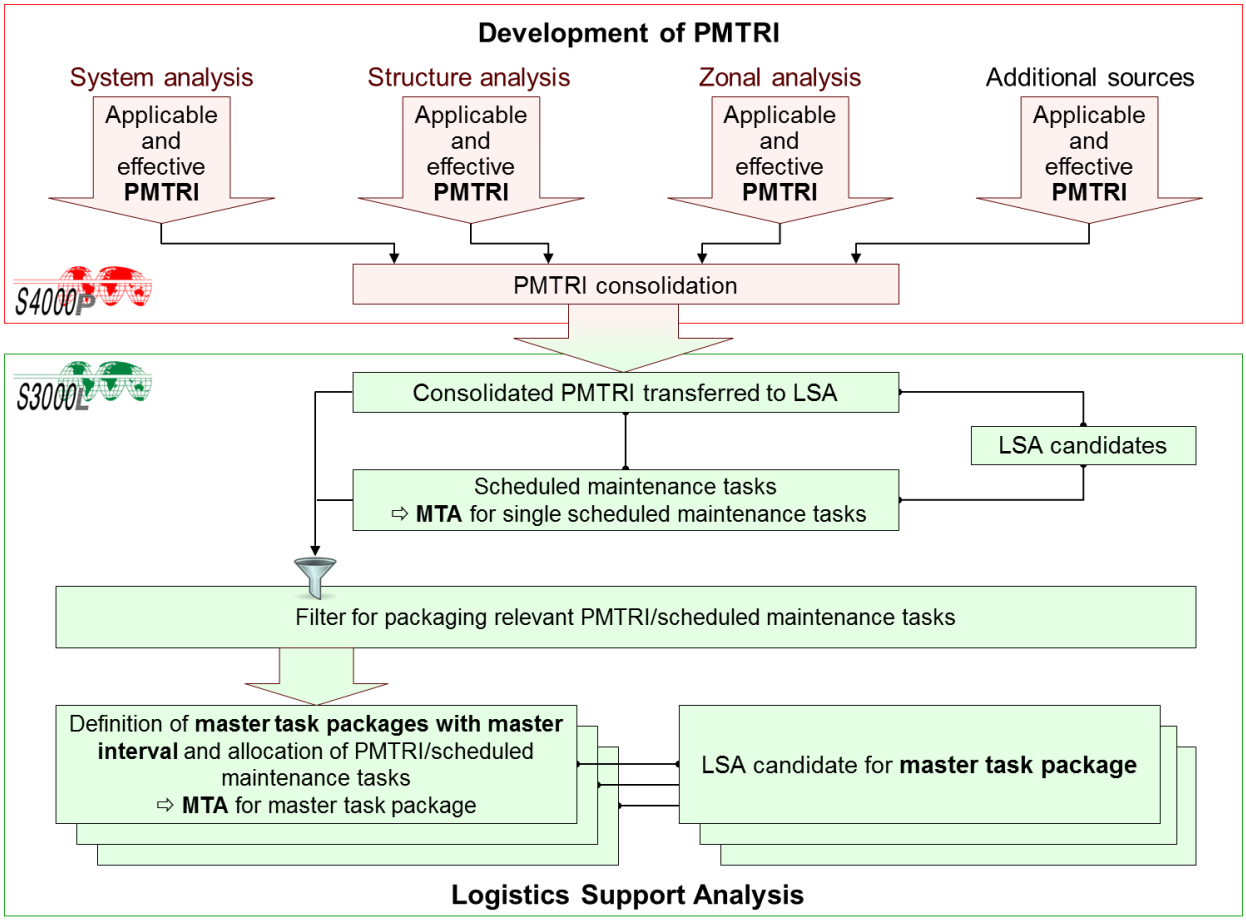
- PMTRI and criticality allocated to Product safety or not complying with the law or environmental integrity ⇨ immediate feedback with urgent need for clarification
- PMTRI and criticality allocated to Product operational/mission availability or best economic feasibility ⇨ feedback and request for clarification depending on contractual requirements between user and manufacturer

4 Packaging for PMTRI

4.1 Packaging process overview

Fig 6 shows an overview of the entire process starting from PMTRI identification within an analytical process (eg, based on S4000P) to a final harmonization in form of adequate master task packages within a PMP. The figure illustrates which activities belong to development of PMTRI (eg, based on S4000P) and those that fall within the scope of LSA. The general process steps are:

- developing applicable and effective PMTRI
- assigning the PMTRI to corresponding LSA candidates within the Product breakdown
- assigning scheduled maintenance tasks to each harmonized PMTRI and a corresponding LSA candidate
- performing an MTA for each scheduled maintenance task
- selecting all PMTRI/scheduled maintenance tasks relevant for packaging
- defining master task packages with corresponding master intervals
- assigning PMTRI/scheduled maintenance tasks to the master task packages
- defining appropriate LSA candidates within the Product breakdown for the allocation of master task packages
- assigning master task packages to appropriate LSA candidates
- performing an MTA for the complete master task packages



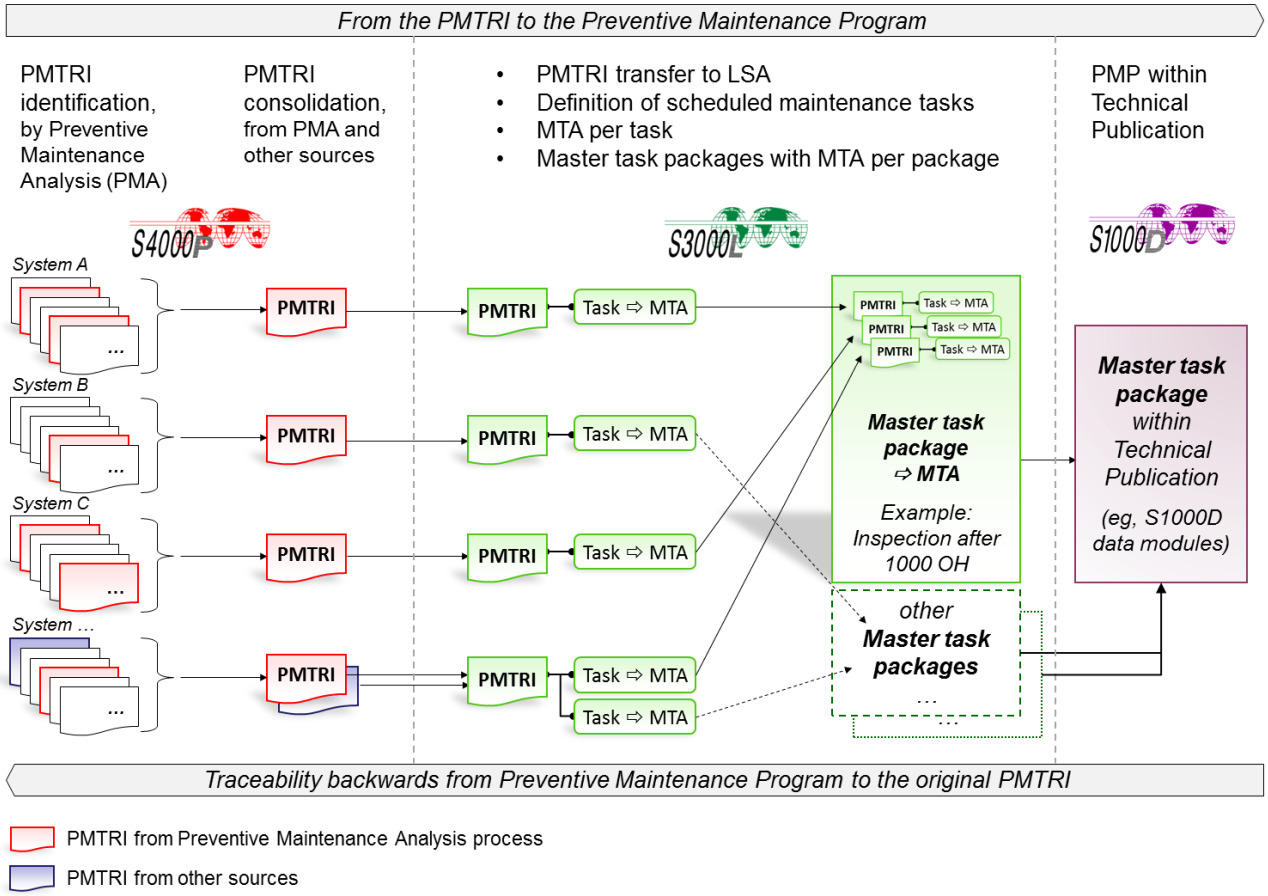
ICN-B6865-S3000L0066-004-01

Fig 6 Process overview from PMTRI to master task packages

Another aspect is traceability of a PMTRI. Fig 7 shows the complete traceability path, from the original PMTRI (eg, developed in accordance with S4000P), via scheduled maintenance tasks and final master task packages developed within the LSA process, to the final master task packages documented within the corresponding technical publication. Finally, the master task packages comprise the binding preventive maintenance activities to be performed during the Product in-service phase.

Note

It is possible to apply the In-Service Maintenance Optimization (ISMO) process described in S4000P to optimize preventive maintenance at a later stage. For this purpose, correct and complete documentation of the entire analysis chain shown in Fig 7 is of crucial importance.



ICN-B6865-S3000L0067-004-01

Fig 7 From single PMTRI to completed PMP (and traceability back)

4.2 Preparation of packaging process

4.2.1 General aspects

In order to start an appropriate packaging process for all identified PMTRI, the Product manufacturer must request information form the customer/operator whether the Product is operated within the usage parameters determined by Product design and development. Additionally, it is necessary to determine and document the process and the project-specific rules for PMTRI packaging, for example, using a corresponding packaging guidance document. It is recommended that the guidance document for the PMTRI packaging process detail at least the following aspects:

- baseline usage scenario and potential deviations for specific customers/operators
- rules for the exclusion of a PMTRI/scheduled maintenance task from packaging
- decisions/predictions about interval unit of measure and parameter values for the master task packages
- calculation rules for adaptation of interval unit of measure, if required
- rules for PMTRI allocation to a master task package
- responsibilities and degree of involvement of regulatory authorities

All resulting PMTRI according to Para 2.1 must be considered by the packaging process within the scope of LSA activities. MTA must analyze the corresponding scheduled maintenance tasks to describe the task execution, determine the Maintenance Level (ML) and identify the required resources, such as personnel, support equipment, spares, consumables, technical publication, facilities and infrastructure. Based on MTA results, it is possible to decide whether an individual PMTRI is actually a candidate for packaging. Any PMTRI leading to a scheduled maintenance

task that can be performed without any preparation work, any additional support equipment, and more or less at any time, is not considered as being relevant for packaging.

It is necessary to define the master intervals (unit of measure and numerical values) of the relevant master task packages after determining the packaging relevant PMTRI and corresponding scheduled maintenance tasks, and confirming Product configuration validity and completeness. In general, the customer/operator needs to be involved in the determination of the master intervals.

For individual Products, it can be necessary to define multiple packaging concepts, possibly with different units of measure for each master task package (eg, based on calendar versus based on Product usage), with a view to allowing different customers/operators to provide an individual definition of master task packages. If a customer intends to deviate from a baseline Product usage scenario, all calculation parameters used to convert intervals from one unit of measure to another must be checked and, if necessary, adapted.

4.2.2 Master task package

Each master task package requires a master interval. The master interval can be based on:

- Product usage parameters
- calendar-based parameters
- combination of Product usage and calendar-based parameters

Examples for Product usage parameters are:

- operating hours
- operating cycles
- number of activations
- driven distance (eg, for vehicles)

Note

If calendar-based master intervals are selected, the actual intensity of usage of the Product can vary with no impact on the scheduled Product maintenance. The actual intensity of usage is better represented by usage-oriented master intervals. For example, the real Product usage can differ significantly depending on individual users.

Depending on the individual PMTRI, more than one threshold can be applicable and effective, refer to Para 2.2.6. In those cases, the threshold expected to be reached first becomes the relevant one for the master task package.

4.2.3 Interval correlation between PMTRI and master interval

It is necessary to correlate the units of measure for the intervals of single PMTRI and the relevant master interval in case they differ. For example, a single PMTRI is based on operating cycles, and the master interval is based on operating hours. In this case, the usage scenario must determine the expected number of operating cycles within a certain number of operating hours (eg, expected aircraft landings within 1000 operating hours). If the correlation is clear, it is possible to describe the interval of the PMTRI using the unit of measure of the master interval. However, this type of correlation always assumes there is a tolerable level of constant Product usage. Peaks in the usage of a Product always cause a risk of exceeding thresholds earlier as allowed, especially in the case of correlating units of measure for PMTRI intervals and master intervals. Also refer to the interval adaptation example in Para 4.3.5.

4.2.4 Preparation completion

Prior to starting the PMTRI packaging process, it is necessary to check the underlying LSA data to ensure all PMTRI are approved and released for further interval packaging activities. The PMTRI must represent all valid Product configurations, including variants of the Product under analysis.

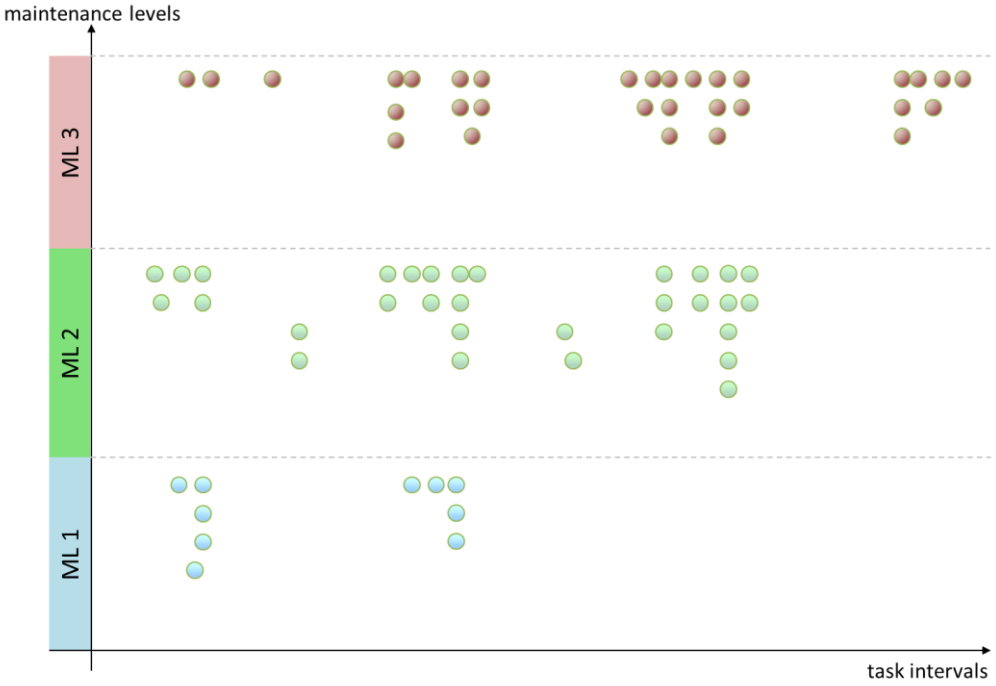
The preparation process aims to ensure an effective packaging of the PMTRI taking into account at least the following aspects:

- A PMTRI can become relevant in more than one master task package (eg, a preventive maintenance task satisfying a PMTRI to be performed prior and after each Product operation day)
- For each identified effective and applicable PMTRI, the "worst-case" criticality of the FFE at Product system level must be available. This criticality category influences decisions in terms of interval extension or reduction to include a PMTRI and the corresponding scheduled maintenance task into a master task package.
- A PMTRI can have one or more different interval types (eg, based on operating hours, cycles, calendar time, and distances) and different numerical interval values. This requires the use of conversion factors to adapt PMTRI intervals and allocate them to master task packages.
- A PMTRI can be relevant only for specific ML, because:
  - the PMTRI can require specific training and experience of maintenance personnel which are only available at specific ML
  - the PMTRI can require material resources (eg, support equipment, spare parts, consumables, facilities and/or infrastructure) which are only available at specific ML

4.3 Packaging process for PMTRI

4.3.1 Impact of maintenance level on master task packages

After packaging preparation is complete, a set of PMTRI with different interval values based on the master intervals is identified, consolidated, and documented within the LSA data. The PMTRI justify scheduled maintenance tasks assigned to different ML after the MTA. Fig 8 provides a generic example for a first distribution of PMTRI.



ICN-B6865-S3000L0110-003-01

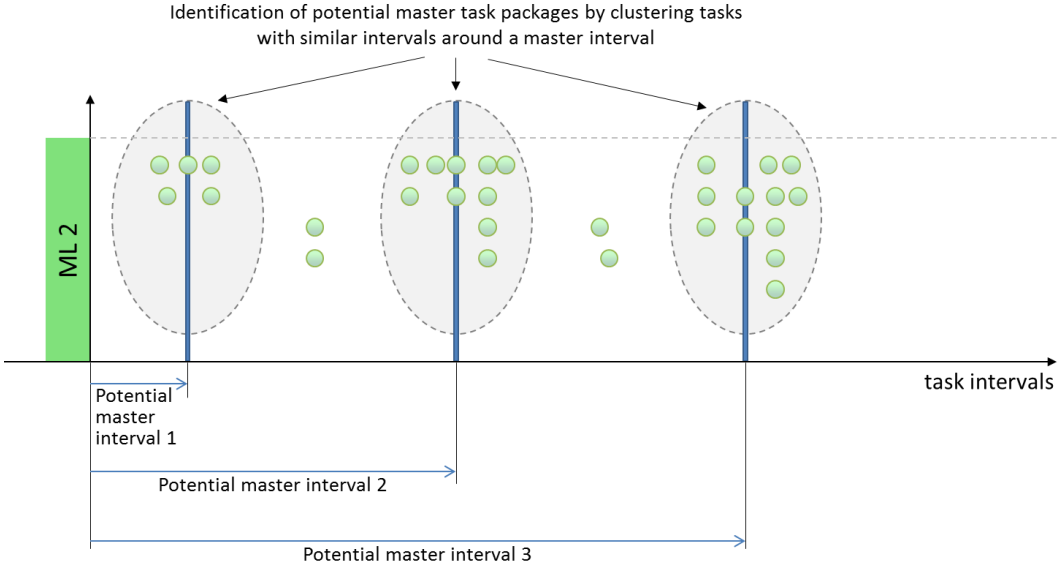
Fig 8 Initial distribution of PMTRI (on different maintenance levels)

Each bullet in Fig 8 represents one PMTRI to be performed at the determined ML.

Note

Chap 11 provides examples of ML definitions.

As early as the beginning of the packaging process, PMTRI distribution within one ML can show a concentration of some PMTRI within limited ranges of intervals. The analyst can use these ranges of similar intervals to support the identification of master intervals. Refer to Fig 9.



ICN-B6865-S3000L0112-002-01

Fig 9 Clustering of PMTRI to support the identification of master intervals

4.3.2 Predefined master task packages

LSA data can also be used and evaluated for all PMTRI/preventive maintenance tasks which are generally determined (eg, for each Product operating day).

In many cases, there are three predefined master task packages for PMTRI allocation:

- Before starting the Product operational/mission phase (eg, preflight inspection of an aircraft)
- After an intermediate stop of Product operation or mission (eg, prior to restart, if applicable)
- At the end of a Product operation or mission (eg, post-flight inspection of an aircraft)

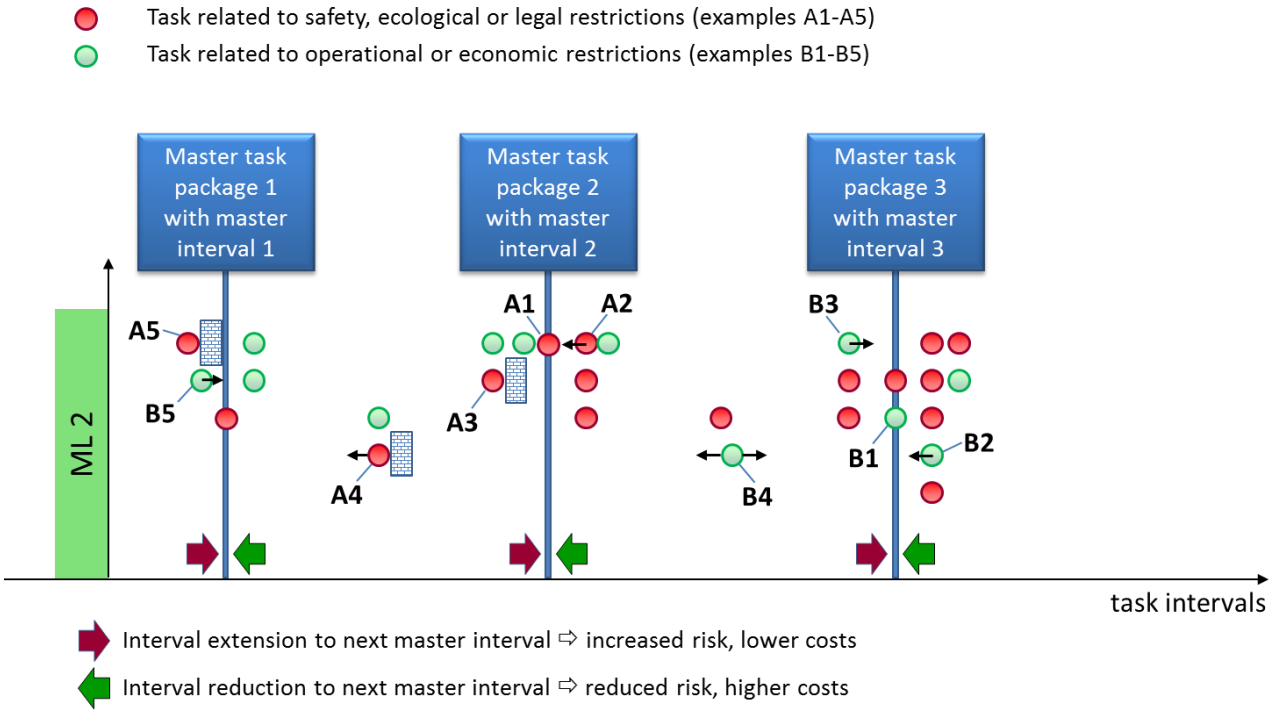
In addition, further predefined master task packages allow PMTRI allocation. For example, the Product overhaul/depot-level maintenance must occur after 15 years and the Product usage period is 30 years. That covers all PMTRI in LSA with a 15-year interval or with an allocation to Product overhaul/Depot Level Maintenance (DLM).

4.3.3 Basic rules for interval adaptation based on PMTRI criticality

Any change in numerical interval values must follow clear rules. These rules must be traceable for regulatory authorities and/or Quality Assurance (QA) departments of the Product manufacturer. For this purpose, it is recommended that adaptation rules be determined, for example, within a project-specific guidance document for the packaging process, which must be accepted by the involved parties.

The adaptation of individual PMTRI must be performed after determining the ML and the master intervals for each master task package. Fig 10 provides examples of use cases for different interval change options. The general consequences to take into consideration are:

- interval extension increases the risk of failure, but reduces maintenance costs
- interval reduction reduces the risk of failure, but increases maintenance costs



ICN-B6865-S3000L0047-002-01

Fig 10 Principles for interval modification

Some PMTRI can have the same numerical value as the master interval. However, most of the interval values of PMTRI typically need to be extended or reduced to fit into a master interval. It is important to remember that extension of intervals normally causes a higher risk of failure.

The examples in Fig 10 consider the different criticality of the corresponding most critical FFE for a PMTRI, and the different needs for adaptation (extension/reduction):

- The examples A1 to A5 are relevant for PMTRI that prevent FFE related to Product safety, compliance with the law or environmental integrity
- The examples B1 to B5 are relevant for PMTRI that prevent FFE related to operational/mission availability or economic feasibility

For each example, adaptation rules are described. Refer to Table 2.

Table 2 General interval adaptation rules

Example	Adaptation rules
A1	Interval of PMTRI equal to the master interval, no adaptation required
A2	Interval of PMTRI higher than the nearest master interval and within a range close to the nearest master interval. Adaptation of interval is possible and agreement by regulatory authorities is expected. High probability of cost increase within acceptable limits, user involvement is recommended.
A3	Interval of PMTRI lower than the nearest master interval and within a range close to the nearest master interval. Adaptation of interval is not allowed because of the related FFE criticality.

Example	Adaptation rules
A4	Interval of PMTRI outside a close range to the next higher or lower master interval. Only adaptation to a lower master interval is possible. Agreement by regulatory authorities is expected. Increase of costs due to a significantly lower interval of task must be considered, user involvement is recommended.
A5	Interval of PMTRI lower than the nearest master interval and within a range close to the nearest master interval, which is the lowest master interval available. Direct adaptation of this interval is not allowed because of the related FFE criticality. This situation requires additional analysis to identify a lower master interval (eg, daily inspection). In addition to that, the safety department can recalculate the interval or, in the worst-case scenario, consider Product design change.
B1	Interval of PMTRI equal to the master interval, no adaptation required
B2	Interval of PMTRI higher than the nearest master interval and within a range close to the nearest master interval. Adaptation of interval is possible. High probability of cost increase within an acceptable limit, user involvement is recommended.
B3	Interval of PMTRI lower than the nearest master interval and within a range close to the nearest master interval. The related FFE criticality allows adaptation of the interval, but the user must agree to the increased risk for FFE.
B4	Interval of PMTRI outside a close range to the next higher or lower master interval. Adaptation to either a lower or a higher master interval possible. In case of interval reduction, it is necessary to consider and the user must agree to the cost increase due to a significant lower interval of task. It is necessary to consider an increased risk of FFE occurrence because of a significant higher interval. User agreement is required.
B5	Interval of PMTRI lower than the nearest master interval and within a range close to the nearest master interval, which is the lowest master interval available. Adaptation to the lower master interval is possible. It is necessary to consider and the user must agree to an increased risk of FFE occurrence because of a higher interval of task.

4.3.4 Summary of PMTRI interval adaptation

It is prohibited to extend intervals of PMTRI that prevent FFE related to safety, compliance with the law or environmental integrity. If any PMTRI interval of these FFE categories, is below the lowest grouping interval for the specific maintenance level, the corresponding tasks needs further investigation, which can result in:

- introduction of a new master task package with a lower interval value
- PMTRI modification (change of scheduled maintenance task to cover PMTRI including recalculation of interval value)
- ML change
- Product design change requirements

It is possible to either extend, or reduce intervals of PMTRI that prevent FFE related to operational/mission availability or economic impact. The user must agree to any modification. It is necessary to balance risk of economic loss or additional downtime with potential cost savings.

Note

The ISMO process described in S4000P takes into account these rules based on the FFE category for each individual PMTRI. The Product in-service experience, in combination with

ISMO, allows an easier adjustment and optimization of all PMTRI linked to FFE on operation, mission and economy.

For safety FFE, compliance with the law and/or environmental integrity, no in-service data/experience can be expected. The optimization of those PMTRI must consequently be based on other investigations, calculations and/or further analysis.

4.3.5 Adaptation example

A safety relevant PMTRI requires a visual inspection on a hydraulic actuator every 600 Operating Hours (OH). The usage scenario is:

As per the Product design, the maximum operating hours for the Product is 1000 OH in one calendar year. The user determines the interval type for the master task package based on calendar, with a small inspection package every 6 months and a full inspection package every calendar year. Due to the PMTRI category "safety-relevant" in this example, it is only possible to reduce the parameter value of the scheduled interval (refer to the rules described in Para 4.3.3).

Based on this limitation, the first preventive maintenance task must be performed during master task packages that do not exceed the repeat time limit of the visual inspection on a hydraulic actuator (here every 600 OH) and the corresponding calendar intervals. Therefore, the scheduled maintenance task must be performed every six months, as the expected operating hours for half a year is 500 OH. It is not possible to select a task which is performed once a year and related to the maximum of 1000 OH, because of a clear interval excess for the safety relevant PMTRI.

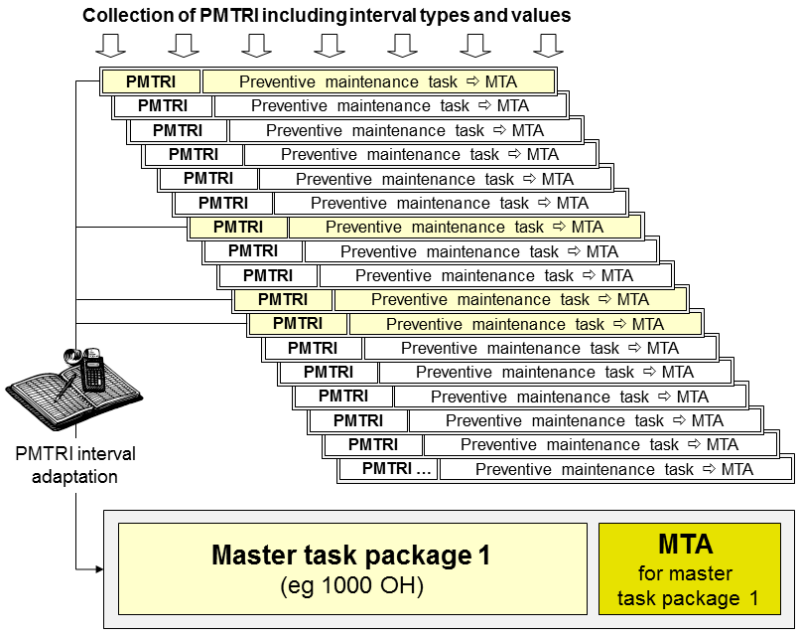
In this example, the original PMTRI for the LSA candidate hydraulic actuator is a detailed visual inspection on the hydraulic actuator every 600 operating hours. The result after the packaging process and corresponding interval adaptation is a detailed visual inspection on a hydraulic actuator every six months performed within the corresponding master task package.

Note

The example also shows a potential conflict with the definition of master task packages with calendar-based intervals. The original PMTRI is based on actual usage (eg, 600 OH), and the master task package is due after six months. The predicted usage of 500 OH per six months is only valid if there is a tolerable level of constant Product usage. The original 600 OH of the safety relevant PMTRI not being exceeded after 6 months of Product operation, must be guaranteed.
The choice often falls on calendar-based master task intervals because they provide a better planning reliability concerning the use of facilities/infrastructure and personnel. Usage-based master intervals, on the other hand, exclude the risk of exceeding critical intervals.

4.4 Maintenance task analysis for master task packages

After allocation of all PMTRI to master task packages, an MTA for the complete packages is required. The master task packages can contain a large number of single PMTRI with or without adaptations of interval. Refer to Fig 11.



ICN-B6865-S3000L0048-003-01

Fig 11 Integration of single PMTRI into master task packages

Typically, the tasks in the master task package are not performed sequentially. It is recommended to consider the following aspects:

- there is potential interdependence of single activities/maintenance tasks (eg, an inspector must close and confirm a maintenance task before another maintenance task can start)
- performing time-consuming tasks for gaining access only once can result in saving time (eg, after opening panels and doors, all tasks are performed within the corresponding area before closing the panels and doors)
- task packaging can consider the possibility of performing tasks in parallel to ensure a more realistic estimation of the duration and effort of a complete master task package
- required resources such as personnel and support equipment must be harmonized for the complete package and are not just a simple collection from the single preventive maintenance tasks

Once a master task package is finalized, the original PMTRI are no longer the basis for logistic evaluations, because the master task package is now the subject to those calculations. However, it is necessary to keep and maintain the original PMTRI in the LSA data to reuse information in later optimization processes and to ensure full traceability over the entire Product life cycle.

At any stage of the project, it must be possible to recover the original PMTRI documented at the LSA candidate in the LSA data from an individual preventive maintenance task defined in a master task package. Moreover, it must be possible to recover the corresponding PMTRI source from each PMTRI (eg, based on S4000P, MSG-3, RCM, and CMR). Refer to Fig 7.

These traceability requirements ensure the complete consistency of the scheduled Product maintenance, starting from the first PMA steps, followed by LSA activities, up to the final generation of the technical publication for a Product.

In addition, this allows to apply the ISMO process with the aim of continuously optimizing Product maintenance, including the Product's Life Cycle Costs (LCC). Refer to S4000P and Chap 14.

5 Associated parts of the S3000L data model

The documentation of the associated data to this chapter is supported by the following Units of Functionality (UoF), refer to Chap 19:

- S3000L UoF LSA Candidate
- S3000L UoF Task Requirement
- S3000L UoF Task Usage
- S3000L UoF Time Limit

Chapter 11

Level of repair analysis

Table of contents

List of tables

References

Table 1 References

Chap No./Document No.	Title
Chap 3	LSA process
Chap 19	Data model

1 General

1.1 Introduction

Level of Repair Analysis (LORA) is an analytical methodology performed on a list of selected LSA candidates and associated tasks. It takes into consideration customer support and operational requirements, customer support capabilities, product technical information, costs, and legal and environmental constraints to determine an optimized maintenance solution. It defines where each selected item is, for example, removed, replaced, repaired, tested, overhauled, inspected, or discarded.

The LORA process harmonizes technical and cost assessment. It is recommended to perform an Economic LORA (ELORA) in conjunction with a Life-Cycle Cost (LCC) analysis, because the costs for maintenance activities are crucial for the complete LCC. It is necessary to combine technical and commercial aspects in order to achieve proper results for LORA.

LORA is a two-step process:

- Determination of whether the Item Under Analysis (IUA) is a repair or discard item (supporting the repair/discard decision by the customer)
- Identification of the optimum maintenance level for the required maintenance tasks, also called ELORA

Note

Maintenance levels are different due to eg, required personnel competence, facilities or support equipment available. Although typically associated with specific organizations and/or geographic locations, the individual maintenance levels in their purest form also denote differences in inherent complexity of maintenance capability. There are several generic levels, for example:

- Organizational level (inspections, servicing, handling, preventive and corrective maintenance)
- Intermediate level (assembly and disassembly beyond the capability of the organizational level)
- Depot level (any action that requires extensive industrial facilities, specialized tools and equipment, or uniquely experienced and trained personnel not available in lower-level maintenance activities)

The LSA Program Plan (LSA PP) or similar documents describe the requirements for LORA. Depending on the specific phase of a project, it is possible to perform LORA for different purposes and using different methods.

The LORA process can be repeated as often as required during the design and development and the in-service phases. LORA results determine whether changes to the maintenance solution are necessary due to design changes or other factors. The results of the LORA process can illustrate various options useful for supplier selection and lead to the adoption of phased support.

1.2 Purpose

The purpose of a LORA is to guide the customer in the selection of the most suitable maintenance solution. Therefore, LORA is performed from a customer point of view, with the customer prerequisites and interests in focus. Different customers can come to different decisions depending on their specific project needs. In the early stage of a project, it is necessary to consider LORA aspects to influence the design (eg, testability features, modularity, and accessibility requirements), as well as to support customer strategy decisions (eg, 2-level maintenance strategy, single source principle).

1.3 Scope

The target readers of this chapter are contractor and customer supportability personnel who perform LORA activities. Appropriate LORA reports will document LORA results. It is recommended to document the derived maintenance solution per LSA candidate.

2 Aspects concerning selections of level of repair analysis candidates

2.1 Candidate selection

The identification of potential LORA candidates can be based on the LSA Candidate Item List (CIL) established in accordance with the project's selection process of LSA candidates. Refer to Chap 3.

The first step is to decide whether an LSA candidate is potentially repairable. If so, it is necessary to choose the most suitable LORA method:

- Best engineering judgment only
- Simplified LORA (eg, collecting information and considering values)
- LORA based on mathematical models (with the aid of commercial software packages)
- Simulation (with the aid of commercial software packages)

It is necessary to document the reason for choosing a specific method. In addition, other constraints or specific customer decisions can include:

- exclusions from repair due to danger of destruction or hazardous/unsafe situation
- input from system integrator or Original Equipment Manufacturer (OEM) relevant to LORA
- cutoff values or design-driven influences (a set of predefined cutoff values can reduce the number of ELORA candidates). These cutoff values are usually based on technical and economic factors such as low task frequency, low unit price, or repair costs.
- customer-imposed maintenance constraints, not related to LORA
- customer's existing maintenance facilities illustrated in the Operational Requirements Document/Customer Requirements Document (ORD/CRD)
- customer requirements derived from ORD/CRD
- contractual requirements or LSA Guidance Conference (GC) decisions
- clear understanding of LORA requirements set by other disciplines

These external influences or requirements must be addressed, discussed, and agreed during the LSA GC.

2.2 Repair or discard decision

A repair/discard decision is the prerequisite to any ELORA. It can include non-economic as well as economic factors. It is, in effect, a screening process aimed at minimizing the number of ELORA candidates using a logical process of elimination. Each LSA CIL has several items that are potential ELORA candidates. Taking a repair/discard decision is necessary to identify which of these items are candidates for further ELORA. This is achieved by determining the repair feasibility for each item.

Below is a list of crucial technical questions for each potential ELORA item:

- Does the design of the item allow repairs in general? If so, this item is a potential ELORA candidate.
- Is the repair of the item limited to certain maintenance levels? If so, this item is a potential ELORA candidate. The analysis must only include the possible maintenance levels.
- Does the design of the item not allow repairs in general? If so, this item must be removed from the ELORA candidate list. However, the proper maintenance level for replace and discard must be identified and documented.

It is necessary to document this decision, and the logic behind it for future traceability. For items that have been identified as potentially repairable, the next phase must determine the method for performing ELORA. This phase provides the opportunity to take into account any external influences/requirements (eg, existing maintenance solutions used by the customer, cutoff values, or influences driven by design).

3 Identification of optimum maintenance level by economic level of repair analysis

The repair/discard decision process results in a list of remaining ELORA candidates identified as potentially repairable. ELORA aims to determine the optimum maintenance solution for these candidates considering mainly economic cost factors such as spare parts, support equipment for failure detection/localization and repair, personnel, training, facilities, technical documentation, and Packaging, Handling, Storage and Transportation (PHST). Additionally, trade-offs should be taken into account when identifying optimum maintenance location. For these trade-offs, non-economic factors such as availability requirements, customer capabilities, reliability, and maintainability can be considered.

Many different mathematical ELORA models are available and supported by different commercial software packages. Simplified LORA often uses an alternate approach that must develop a project specific or tailored model that addresses the requirements and constraints (for both customer and OEM) such as available infrastructure, personnel, or technical capabilities.

Within a project, all LORA analyses must be integrated into a common ELORA model as much as possible. This allows an overview of the cost versus performance of all selected LORA solutions.

4 Data gathering for level of repair analysis

The following sections provide additional details regarding data needed for LORA. Some of them, cost data in particular, are not identified in the S3000L data model yet. Refer to Chap 19.

4.1 Definition of operational concept

Apart from technical and economic data, the LORA outcome depends on the Product intended context of use. These prerequisites are defined in a CONcept of OPerationS (CONOPS), which defines Product distribution and use. A CONOPS is normally provided by the customer. The ORD should contain all necessary information derived from the CONOPS.

4.2 Determine parts hierarchy

Maintenance and operational support activities in LORA are assigned to the hierarchical breakdown structure of the Product systems. Failures at the top level of a system are assumed to be primarily caused by failures of the items (or subsystems) at the next lower indenture level. Similarly, failures of the items in the second level of indenture are mainly initiated by failures of the items in the next lower indenture level. For this reason, it is impractical to perform a LORA until hierarchical relationships are outlined. For Products that are currently in production, existing LSA data derived from the Product breakdown can determine the hierarchy and subsequent LORA candidate items.

4.3 Determine level of repair analysis candidates

After establishing the parts hierarchy, the analyst must determine which parts will be included in the analysis. At first glance, it seems logical to include all of the parts. However, non-repairable and consumable parts (eg, nuts, bolts, gaskets) will add no value to the analysis and, therefore are excluded. Establishing a set of rules based on existing repair or maintenance codes, unit price, repair level, or other criteria can help determining the candidate items.

4.4 Collect unit prices

Unit prices for all items are required to fill an ELORA data model and perform the analysis. Unit prices are available from provisioning records or other documents. The unit price for all ELORA candidate items is necessary to determine the economic feasibility of repair. It is essential to use correct unit prices for ELORA calculation purposes. Unit prices can differ dramatically from the production phase to the in-service phase.

Note

The price of spare parts can be very different from the production unit price.

Sometimes, precise values are difficult to obtain and it is only possible to make an estimate. These estimations can be acceptable during the initial phases, but they must be reviewed and/or evaluated through sensitivity analysis.

4.5 Collect maintenance-related cost data

Maintenance cost data include labor time, training, spare parts and consumables, support equipment, facilities, technical documentation and PHST aspects. With respect to these data, it is necessary to distinguish between initial and recurring costs. These data are used to calculate costs at each possible maintenance level.

Table 2 Overview of typical cost elements provides an overview of typical cost elements to be included in ELORA. Additional data can be required depending on project requirements.

Table 2 Overview of typical cost elements

Cost element	Initial	Recurring
Personnel		
– Labor time costs		X
– Training costs (teachers and staff)	X	X
– Training equipment costs (eg, simulators, classroom equipment, training facilities)	X	X
Note It is recommended that costs of direct labor and indirect labor (eg, line managers, inspectors, work preparation department) be considered.		
Spare parts / consumables		
– Spare parts / consumables provision	X	X
– Spare parts / consumables storage		X
– Disposal costs		X
Support equipment to diagnose and repair		
– Support equipment provision and replacement	X	
– Support equipment maintenance		X
– Support equipment upgrade	X	
– Development of customized software (eg, test software)	X	
– Support of customized software for support equipment		X
– Disposal costs		X

Cost element	Initial	Recurring
Facilities and infrastructure		
– Building of facilities and infrastructure	X	
– Maintenance of facilities and infrastructure		X
– Costs of operation for facilities and infrastructure		X
– Costs for modification of facilities and infrastructure	X	X
Technical documentation		
– Documentation costs (eg, user handbooks, maintenance and training manuals, illustrations, spare part catalogues)	X	X
PHST aspects		
– Packaging costs	X	X
– Stock keeping costs (eg, maintenance tasks and handling during storage, special storage containers, administration of stock)	X	X
– Transportation costs (transport to maintenance facilities and back to user)		X
Others		
– Average repair costs at industry level	X	X
– Additional costs (documentation, administration, preparatory work and post processing)	X	X
– Restocking costs	X	X
– Requisition costs	X	X
– Disposal costs	X	X
– Discount rate	X	X
– Holding cost percentage	X	X
– Interest rate	X	X

4.6 Other maintenance relevant data

Besides data on direct costs, other pieces of information can be useful for the ELORA process. This information includes but is not limited to:

– Task execution information

- Task complexity
- Required support and test equipment and required competence to use it
- Legal regulations which influence the task execution
- OEM recommendations or restrictions because of warranty

– Frequency and duration information

- Reliability information, eg, Mean Time Between Failure (MTBF), Mean Time Between Unscheduled Removal (MTBUR)
- Frequency of corrective maintenance task
- Preventive maintenance interval
- Task duration
- Logistics down time (eg, waiting time for support resources)

Note

Data for repair duration are often difficult to obtain, particularly for newly developed equipment or systems. During development stage, desired and maximum duration for failure repair can be specified. This can also serve as a repair duration for items directly removed and replaced. For estimation of repair duration of lower indenture items, it can be necessary to use the repair duration for similar items in other systems.

– Availability values

- Minimum availability of IUA (usually, operational availability is specified)
- Availability of personnel for maintenance activities
- Availability of spare parts in stock
- Availability of support equipment
- Availability of facilities and infrastructure

– Stock relevant values

- Procurement lead time
- Pipeline transit times
- Repair turnaround times

– General information

- Number and type of contractor industrial facilities
- Number of operational sites
- Number of operated Products per site
- Distances between sites
- Inflation/discount rate
- Repair policy (eg, 2-level maintenance strategy)

It has been noted that the majority of decisions that influence costs must be made at a very early stage, even if the available information is limited and the design is not stable yet. For this reason, it is necessary to perform LORA throughout the Product life cycle, to identify a potential need for updating the current maintenance solution in a timely manner.

5 Performance of economic level of repair analysis and preparation of results

ELORA is performed after collecting and harmonizing data together with the customer. A simple approach must evaluate and balance the given information using best engineering judgment. It is possible to select a commercial software package in case more complex mathematical calculation models are used. The data set to be collected depends on the requirements of the specific software package. The results of the calculation runs must be laid out in an ELORA report and distributed according to established rules.

5.1 Economic level of repair analysis baseline run

The baseline run using the gathered ELORA data is the first step to perform an ELORA using a supporting software package. This baseline run will result in a first set of information regarding costs for the individual maintenance levels. Depending on the different maintenance levels, the result can be either clear-cut or ambiguous. In either case, it is recommended to confirm the baseline run results by executing a sensitivity analysis.

5.2 Sensitivity analysis

It is necessary to perform sensitivity analysis on various parameters that are cost significant. The results of the various sensitivity runs will indicate whether the maintenance solution from the baseline run is stable or not. Sensitivity analyses can include, but not limited to:

– major cost influencing parameters of the IUA itself (eg, unit price, MTBF)

- any parameters that are critical to costly support requirements (eg, facilities, infrastructure, support equipment, highly educated personnel)
- any parameters using assumptions or estimates because of unavailable data (eg, historic or similar data from other Products)

Once the parameters for sensitivity analysis are selected, the next step must establish an appropriate numerical range for the parameter. Finally, when all the requirements are established, ELORA is performed for each parameter variation. Changing single parameters will keep the number of runs to a minimum. Multiple parameter changes will lead to nested analyses. Therefore, it is recommended that these changes be avoided, as they may cause confusion while evaluating the results.

The output of the sensitivity analysis can be used as a baseline to establish a preliminary maintenance solution. Additionally, it is useful to influence other support disciplines and/or design, in order to change their input with a view to achieving the expected maintenance solution for the equipment.

A sensitivity analysis is usually conducted to complement baseline results. Sensitivity analyses are performed for cost significant parameters that have a low confidence level or to accomplish a trade-off study. In case of sensitivity analysis of low-confidence values, the most common method recommends analyses using the worst- and best-case data. If there is no shift in the results of the maintenance solutions in the two analyses, then no additional executions using intermediate values are required.

An example is sensitivity analysis for an item's MTBF. In this example, the baseline analysis uses an estimated value of 5,000 operating hours. Preliminary analysis reveals that MTBF for similar items range from 1,500 to 15,000 operating hours. The first sensitivity analysis would include 1,500 and 15,000 operating hours for the MTBF of this item. If the most cost-effective repair level for both sensitivity analyses is the same as the baseline analysis, no further sensitivity analysis for this MTBF is necessary. However, if a change in repair level occurs for one or both analyses, further analyses are necessary and must use MTBF values between the highest and lowest value to identify the break points.

5.3 Maintenance solution recommendation

To achieve a concluding maintenance solution recommendation, it is necessary to consider costs resulting from ELORA, as well as technical aspects that can have a major impact on the recommendation of the maintenance solution to the customer (eg, destruction of the IUA during critical repairs, dangerous repair tasks). Additionally, it is necessary to consider the customer's preferences.

It is also important to harmonize the maintenance activities in order to avoid splitting similar maintenance activities to different maintenance sites. For example, it is recommended that the allocation of a piece of electronic equipment for replacement and subsequent repair of different components be not split between industry and customer sites.

5.4 Level of repair analysis report

It is necessary to produce a report documenting the analysis results. The LORA report must contain at least:

- agreements reached between the customer and/or the contractors regarding LORA
- a brief description of the system/equipment under analysis
- comprehensive list of any assumptions and estimations, including their rationale
- data sources
- comprehensive list of the LORA data elements
- input values
    - Operational environment data used for the process
    - Breakdown of IUA

- Cost and maintenance data

– mathematical methods applied in a simplified ELORA
– use of ELORA software package
– results of baseline run
– results and explanations of sensitivity analysis runs
– trade-off studies
– consolidation of technical and cost aspects
– concluding recommendation on the maintenance solution, including discard decision

It is recommended that a report structure and layout be developed early in the analysis process. Otherwise, there is the risk of overlooking important information and logic during a lengthy analysis.

For good traceability, it is recommended to use a standardized format for the maintenance solution recommendation and related customer decision within the LORA report. The customer decision will be documented appropriately.

5.5 Maintenance solution documentation

The content of the maintenance solution relates to the corresponding maintenance activities, as documented in the description of maintenance tasks. Each task for an LSA candidate must include attributes that reflect the maintenance solution. This information must include at least:

– maintenance task type (eg, preventive or corrective)
– interval or threshold data for preventive tasks
– the maintenance level at which the task must be performed
– the place at which the maintenance tasks must be performed
– special remarks or warnings (eg, if the Product is used under special conditions, the inspection interval must be reduced from 6 to 3 months)
– references to standard repair tasks
– data source information for traceability (eg, identification of LORA report)

It is recommended that the basic data set reflecting the maintenance concept be agreed during the LSA GC, to ensure mutual understanding. Proper documentation within an appropriate information record ensure to report effectively different aspects of the maintenance concept.

Examples:

– Complete overview of maintenance activities at all maintenance levels
– Summary of maintenance activities performed at certain maintenance levels
– Overview of preventive/corrective maintenance activities
– Expected effort at certain maintenance levels
– Support for quality checks concerning completeness and conclusiveness of the maintenance solution
– Predicted maintenance and support cost distribution according to the defined maintenance solution

6 Example of maintenance level definitions

6.1 General

This example is based on three maintenance levels, and indicates personnel capability, availability of special facilities, time limits, and environmental conditions to consider with a view to determine the tasks relevant to each maintenance level.

6.2 Level 1

Level 1 maintenance aims to ensure the Product remains available. This implies personnel will perform a fast and easy replacement of Line Replaceable Units (LRU) and/or the replacement of modules on the Product when a malfunction occurs.

Level 1 activities typically include, but are not limited to:

- PHST and servicing activities
- preparation for use and role changes
- pre- and post-operation inspections
- functional checks
- trouble shooting
- preventive maintenance
- corrective maintenance (eg, repair by replacement and system adjustment)
- loading of software (operational and engineering) and data retrieval
- simple modifications

6.3 Level 2

Level 2 maintenance aims to maintain the highest possible level of availability. Maintenance activities at the operating site also include the repair of subassemblies, modules and LRUs after their replacement at maintenance level 1. Testing on test-benches or integration tests can be included. Subsequently, level 2 maintenance can be performed either on Product or at specific repair shops.

Level 2 activities typically include, but are not limited to:

- repairs down to module and subassembly level
- moderate structural repairs
- major planned inspections
- moderate modifications
- technical assistance for the organization of Level 1 maintenance
- software servicing concerning engineering data
- preservation of complete Product

Level 2 activities include corrective and preventive maintenance and specific maintenance activities that will be performed on Product (eg, during maintenance, the Product will be unavailable for use), or at specific repair shops. Level 2 also includes tasks not performed at level 1 to facilitate the return of the equipment to its full operational state. Level 2 maintenance will be performed at suitable facilities for the performance of maintenance tasks, which can require the use of special equipment or specialized repair shops, and must be performed by appropriately trained and specialized personnel.

6.4 Level 3

Level 3 maintenance aims to ensure the highest possible availability of the Product, as well as providing engineering support for operational aspects. It must ensure all repairs and overhaul activities beyond level 1 and level 2 capabilities. Major modifications to improve the design and/or operational activities will be prepared and, if necessary, performed at this level.

Level 3 activities are expected to typically include, but are not limited to:

- repairs down to full reconditioning
- repairs requiring special, rare skills or support equipment
- major structural repairs
- major planned inspections
- extended modifications and update programs
- technical assistance for the organization of level 1 and level 2 maintenance
- software modification
- preservation of the complete Product

Level 3 maintenance must ensure the utmost autonomy for the users' organizations. International cooperation can allow the setup of an effective and economic authorized level of maintenance. It is recommended that single-source repair be considered the best solution.

Level 3 maintenance will be performed in duly equipped facilities or component repair facilities of the customer or contractor (or subcontractors). Level 3 requires appropriately trained and specialized personnel. Level 3 maintenance also comprises the return of defective items (suspected or confirmed) to the OEM for repair/overhaul/retest.

Note

As the "highest" maintenance level, level 3 activities can be subdivided into two or more levels (eg, level 3 and level 4). This can be used when required to separate clearly user-operated activities (level 3) from contractor-operated activities at industry facilities (level 4). In this case, the maintenance strategy contains four levels of maintenance (or possibly more).

7 Associated parts of the S3000L data model

The documentation of the associated data to this chapter is supported by the following Units of Functionality (UoF), refer to Chap 19:

- S3000L UoF Breakdown Structure
- S3000L UoF Hardware Element
- S3000L UoF LSA Candidate
- S3000L UoF LSA Failure Mode Group
- S3000L UoF Part Definition
- S3000L UoF Performance Parameter
- S3000L UoF Task Requirement
- S3000L UoF Task Usage
- S3000L UoF Time Limit

Chapter 12

Task requirements and maintenance task analysis

Table of contents

List of tables

List of figures

References

Table 1 References

Chap No./Document No.	Title
Chap 7	Corrective maintenance analysis

Chap 8	Damage and special event analysis
Chap 9	Operational support analysis
Chap 10	Development of a preventive maintenance program
Chap 11	Level of repair analysis
Chap 13	Software support analysis
Chap 16	Disposal
Chap 19	Data model
Chap 21	Terms, abbreviations and acronyms
Chap 22	Data element list
S1000D	International specification for technical publications using a common source database

1 General

1.1 Introduction

For a complete documentation of all support activities (generally referred to as *task* in this chapter) in the context of corrective maintenance, preventive maintenance, and operational support, it is necessary to consider two main aspects:

- each rectifying task (refer to Para 5.1) needs to be justified by at least one corresponding task requirement
- each task needs to be analyzed in terms of procedure (how the task is executed) and personnel/material resources required to perform the task

Creating a task requirement within the LSA data represents the justification of a task. The typical task requirements are identified by a set of analysis activities described in Chap 7, Chap 8, Chap 9, Chap 10, Chap 13, Chap 16, or by other additional analysis activities.

Task analysis must take into account different aspects:

- organizing the task into subtasks (also often referred to as work steps)
- including all steps necessary to prepare and restore the Item Under Analysis (IUA)
- describing the procedure how to perform the task
- identifying task resources
- documenting additional relevant information for each task (eg, task duration, task location, safety criticality for personnel, operational impact)
- identifying the maintenance level and frequency of task performance

Note

A task can contain hundreds of subtasks, which can be performed sequentially or within specific parts in parallel. It is necessary to clearly define the sequence and interdependency between the subtasks.

1.2 Purpose

This chapter is a guideline for the analysis of an identified task with regard to its justification and detailed performance, and to the identification of all required resources. Maintenance Task Analysis (MTA) must identify everything that is relevant to the correct task performance.

1.3 Scope

The target readers of this chapter are analysts who perform analysis activities within the LSA process to describe the identified tasks in detail. This includes a task description and the identification of required spare parts, consumables, support equipment, personnel, and facilities/infrastructure. It is necessary to take into consideration additional information such as task duration, task criticality, maintenance level and task location, training needs, pre- and post-conditions or safety requirements.

1.4 Terms, abbreviations and acronyms

Table 2 provides definitions that are specific to MTA. Chap 21 provides the complete set of LSA terms, abbreviations, and acronyms.

Table 2 Terms, abbreviations and acronyms

Term	Definition
Operational support task	An operational support task is any support activity that fulfils a task requirement in the context of Product operation, such as servicing or Packaging, Handling, Storage and Transportation (PHST) requirements.
Rectifying task	A rectifying task is any support activity that resolves an issue, such as failures, damages, special events, or thresholds.
Supporting task	A supporting task is a part of a complete rectifying or operational support task. Supporting tasks alone cannot rectify issues such as failures, damages, special events or thresholds.

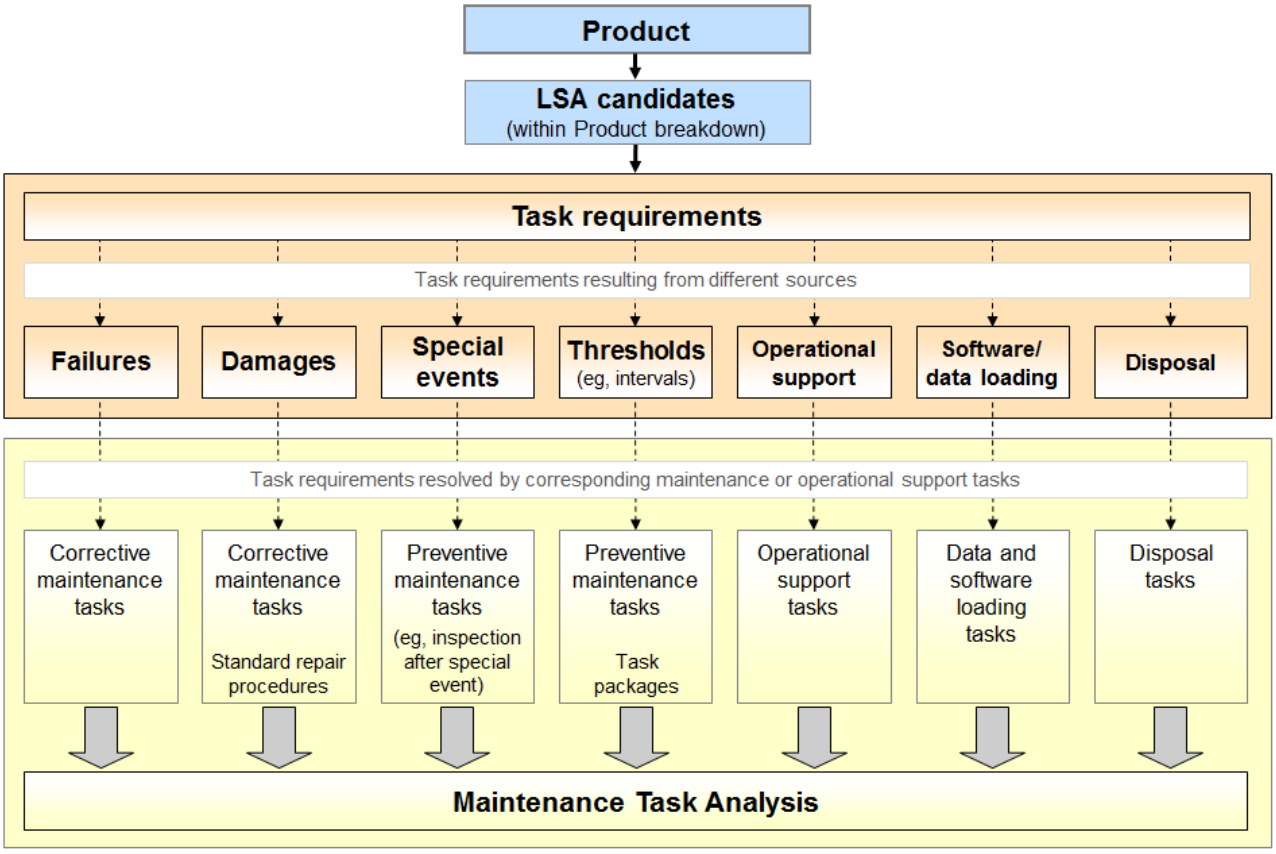
2 Task requirements

A justification illustrating the reasons and triggers for a task is necessary for each task. The principle is to define a task requirement for each rectifying task and operational support task. One of the main aspects is the approach of event-driven maintenance. A task requirement can relate to an event identified by the corresponding supportability analysis activities. Failures, damages, special events, and thresholds are the primary triggers for maintenance tasks, and each has its specific chapter. Refer to Chap 7, Chap 8 and Chap 10. Additionally, some tasks originate from other sources like operational support tasks to cover handling and servicing requirements, software or data loading tasks, and disposal tasks. Refer to Chap 9, Chap 13 and Chap 16. Finally, also directives issued by authorities or manufacturers (often in relation with warranty issues), as well as law regulations (eg, environmental regulations) can justify tasks.

Sometimes, particularly when analyzing operational support tasks, it is not possible to clearly determine it is a maintenance-relevant event or a general support requirement that justifies a task. For example, a task for towing an aircraft can have several different justifications (⇨ task requirements):

- towing out of the hangar to prepare for a mission
- towing for refueling purposes
- towing to the maintenance hangar in case of a failure

For proper documentation of the relationship between the justifying task requirement and the related task, the S3000L data model has established a corresponding data structure. Refer to Chap 19. Fig 1 shows an overview of the relationship between tasks and their associated justification.



ICN-B6865-S3000L0049-003-01

Fig 1 Relationship between task requirements and support tasks

Task requirements can change during the product lifetime. The causes of such changes can be engineering changes, changes of the support environment or usage scenarios. For that reason, both, task requirements and tasks can have several revisions over time in order to keep an appropriate traceability (refer to Chap 19).

3 Categorization of tasks

For task categorization, it is recommended to use a coding system, such as the information code given in S1000D for technical publications (refer to S1000D).

Each information code has a clear definition in S1000D and enables an easier integration between LSA and technical publications using S1000D. This avoids the need for a matrix to convert LSA task coding and nomenclature to the coding and naming used within a technical publications system. This approach improves commonality, traceability, and therefore the quality of IPS.

There are appropriate verbs to identify activities clearly, and it is necessary to use them when defining the information code for a special activity. If a more detailed information is available, avoid the usage of general information codes. Table 3 shows some examples from S1000D that also illustrate the generalization/specialization aspect for information codes.

Table 3 Examples of S1000D information codes usable for task categorization

Information codes			Information code definition
200			Servicing
	210		Fill
		212	Fill with oil
		215	Fill with air
300			Examinations, tests and checks
	310		Visual examination
		312	Examination with a borescope
500			Disconnect, remove and disassemble procedures
	520		Remove procedure
		525	Ammunition unloading

4 Task documentation

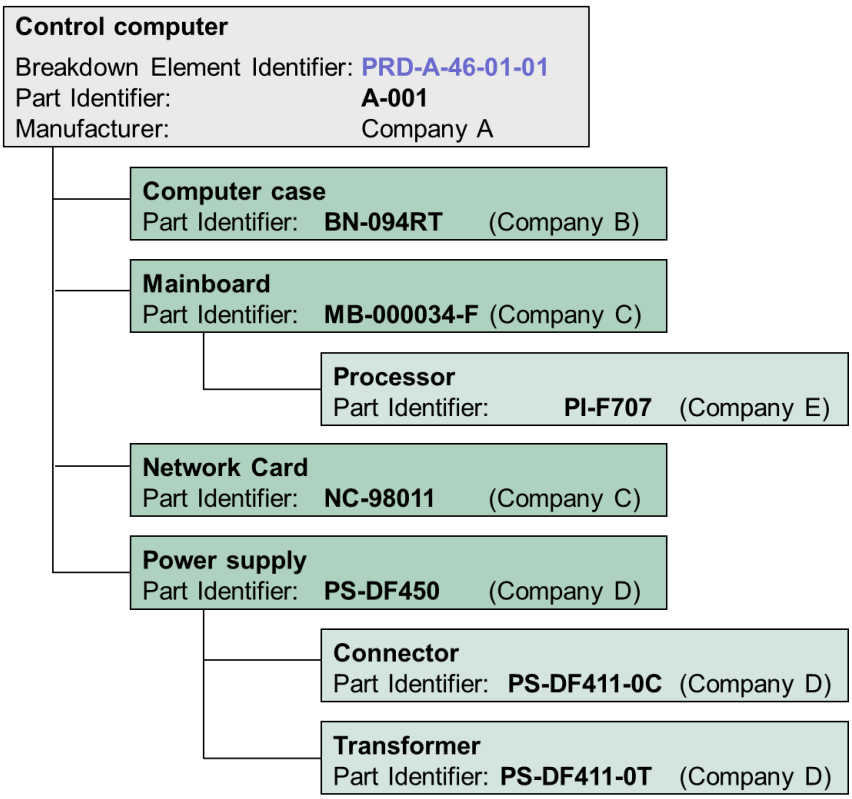
4.1 General aspects

4.1.1 Tasks associated with breakdown elements or parts

Tasks are associated with LSA candidates. As described in Chap 3, LSA candidates can be breakdown elements or parts. This means a breakdown element or a part (identified as an LSA candidate) can have one or many associated tasks. If a task is associated to a breakdown element, this often indicates it is necessary to perform the associated task on the Product. In this case, the installation location is of crucial importance. For example, remove/install tasks are typically associated to a breakdown element, because removal or installation of an equipment always relates to the place where the equipment is located. Tasks associated with a part are often performed on a part removed from the Product (eg, task to be performed on bench in a workshop). In this case, the part is no longer associated with its use/installation within the Product.

4.1.2 Level of task or task requirement association

Any task requirement or task can be associated to different levels of the Product breakdown. The association of task requirements or tasks depends on the depth of Product breakdown and on the concept to associate a repair or a replace task or the corresponding task requirement. Fig 2 provides an example of equipment used in this chapter for further explanation.



ICN-B6865-S3000L0050-002-01

Fig 2 General equipment example

The example shows a simple breakdown of the "*control computer*" equipment. "*Company A*" provides a part identifier for the equipment. It contains a part list with corresponding part identifiers for each part, as provided by different companies.

Note

A part list is often referred to as a Bill of Material (BOM).

Theoretically, the assembled part (eg, the complete computer), but also components from the part list of the assembled part, can meet the criteria to be a potential LSA candidate. From an LSA point of view, there are different possibilities to associate tasks.

Since it is mandatory to consider the documentation of all LSA data associated with an LSA candidate as a cost-intensive activity, it is recommended to restrict breakdown elements and/or parts to be selected as an LSA candidate. Table 4 shows a typical either-or situation of LSA candidate selection and corresponding task association.

Table 4 Task association with different levels of LSA candidates

Assembled part view	Part/component view
Assembled part (complete equipment) selected as LSA candidate ⇨ **one** LSA candidate for task association	Parts from the part list of the assembled part selected as LSA candidates ⇨ **many** LSA candidates for task association

Assembled part view	Part/component view
Consequence: One or many maintenance tasks to be exclusively associated with the assembled part: – repair control computer by replacement of mainboard – repair control computer by replacement of network card	Consequence: One or many maintenance tasks to be associated with many single parts from the parts list: – replace mainboard – replace network card – replace power supply

Note

The examples of Table 4 refer to the equipment example of Fig 2.

LSA candidate selection must follow a common philosophy within a project. The project specific LSA GD must define and explain clearly when to use a repair or a replace task to describe maintenance, as well as determine which breakdown level of the IUA is suitable to become an LSA candidate.

4.2 Practical considerations

In reality, the situation is more complex than what is described in Para 4.1.2. Considering the example in Fig 2, components in the assembled part can be repairable or discard items. Therefore, a repairable part within an equipment can become an LSA candidate as well.

It is fundamental to determine the properties of equipment parts (including the equipment itself) with respect to reparability and maintainability. Table 5 shows typical properties of parts concerning reparability and/or maintainability.

Table 5 Reparability and maintainability properties

Part properties	Consequences for reparability and maintainability
No corrective and/or preventive maintenance	A part that will not be repaired/maintained. In this case, a task requirement/task is only necessary if the required part replacement is performed at customer maintenance level. Replacing the part on or off the Product can determine the element of the Product breakdown associated with the task requirement/task (breakdown element if on the Product, assembly part if off the Product).
Corrective and/or preventive maintenance on industry maintenance level only	An equipment/part that can only be repaired/maintained at industry level. For such items, the detailed description only relates to their replacement (also refer to previous example). It is recommended to identify the possibility of repair and/or any other maintenance (eg, preventive maintenance tasks) on industry level. Usually, a detailed description of any corrective and/or preventive maintenance on industry level within the LSA data is not required.
Corrective and/or preventive maintenance on customer maintenance level	A part that can be repaired at customer level. For such items, it is necessary to document the corresponding task requirements/tasks for both corrective and preventive maintenance tasks. The LSA GC must clearly define level of technical detail for the maintenance tasks.

The properties of a part concerning reparability and maintainability depend on the events influencing the items. There can be failures that lead to the disposal of the complete equipment/part, while other failures will result in the need to repair the equipment/part. Although being aware that a definitive decision is not always possible, it is necessary to analyze each event to identify the relevant maintenance task and maintenance level to resolve the event.

Correction of some events can occur at several maintenance levels. In this case, it is necessary to consider both the types of influencing events and the economic aspects. For example, a Level of Repair Analysis (LORA) can lead to the replacement of the complete equipment (with a follow-on discard) due to economic aspects, although the equipment as such would be repairable. Mixed scenarios are also possible. For example, a certain percentage of a specific failure is rectified at customer maintenance level, and the balance will be returned to the industry maintenance level because of capacity constraints at customer maintenance level. Different deployment scenarios can affect the decision on whether to repair the element due to facility and infrastructure requirements.

Note

The term "reparability strategy" indicates the final detailed decision on the method and location for equipment/ part repair.

In practice, there can be several different solutions concerning events that drive maintenance (especially failures), and the following required chain of corrective maintenance tasks can occur. For better understanding, Table 6 gives an overview of typical examples. Additionally, further examples are given in Para 10.

Table 6 Reparability strategy examples

Event	Corrective maintenance task (rectifying)	Potential follow-on task	Consequences of reparability strategy
Failure of a non-repairable part within an equipment	Repair equipment by replacement of faulty part at customer site	Disposal of faulty part	Non-repairable part is required as a spare part. If the discard of the faulty part requires a description, a specific disposal task will be documented for the non-repairable part (as the appropriate LSA candidate).
Failure of a repairable part within an equipment, faulty part is repairable at customer level (case 1)	Repair equipment by replacing faulty part at customer site	Repair faulty part at customer site	Product is used before the completion of the repair of the faulty part. The part itself is required as a spare part for the repair of the equipment. Components needed to repair the faulty part are required as spare parts for the follow-on task.
Failure of a repairable part within an equipment, faulty part is repairable at customer level (case 2)	Repair equipment by repairing faulty part directly at customer site	None	Product is not used until the completion of the repair of the part. The part itself is not required as a spare part for the repair of the equipment. Components needed to repair the faulty part are required as spare parts.

Event	Corrective maintenance task (rectifying)	Potential follow-on task	Consequences of reparability strategy
Failure of a repairable part within an equipment, faulty part is only repairable at industry level	Repair equipment by replacing faulty part at customer site	Forward faulty part to industry	Repairable part is required as a spare part at operational site. Documentation of repair on industry level is not required.
Major failure of equipment, no repair possible, faulty equipment can be replaced at customer level	Replace equipment at customer site	Discard faulty equipment	Equipment is required as a spare part on customer site. If the discard of the faulty equipment requires a description, a specific disposal task will be documented against the equipment (as the appropriate LSA candidate).

Depending on additional aspects (eg, competences at the different maintenance levels, scrap rates of parts, Product waiting for repair or not), there are multiple ways to influence the sequence of activities. The examples clarify the need to summarize maintenance tasks within a limited number of LSA candidates. Possibly, several follow-on tasks on higher maintenance levels do not require a detailed analysis. The only aspect that can be important is their identification.

In most cases, two task types can resolve equipment failure:

- replacement of the entire equipment (represented by a specific breakdown element)
- repair of equipment by replacement of parts (as a follow-on task)

Another important aspect is that not every replace task can be performed at a customer site. For this reason, it can be necessary to transport the complete Product to industry to replace specific equipment. In this case, a task requirement/task for the transportation of the entire Product is required.

5 Task structure

This paragraph reflects a common understanding about how to create a proper task structure. It is necessary to take into account aspects such as the categorization of tasks, establishing a hierarchy of tasks and subtasks, and reuse of information within already existing tasks with the help of references.

Note

There are different potential solutions to document the content of a task. Documentation of tasks depend on the methodology used by the analyst. For example, it is possible to reduce the repair of an equipment to the pure repair activities just relevant for the equipment already tested and removed from the Product. It means "ready for repair" in the repair shop. However, the repair task (eg, gain access, remove/reinstall equipment, test, undo access) can also include the preparation before the actual repair.

5.1 Task classification

The fundamental approach is to divide the tasks into three different task classes:

- rectifying tasks ("the event solver")
- operational support tasks
- supporting tasks

One or more task requirements drive each rectifying task or operational support task. In the case of a rectifying task, the task justification is typically an event. Such event can be a failure, a damage, a special event, or an expired threshold/time-limit. A maintenance task will resolve these events. Each task that can resolve such an event can be defined as a rectifying task. Typical examples for rectifying tasks are given in Table 7:

Table 7 Typical rectifying tasks for typical events

Event	Rectifying tasks
failure	repair or replace tasks
damage	repair or replace tasks
special event	inspection tasks
threshold	preventive maintenance tasks

The same holds for an operational support task, which typically resolves a concrete task requirement in the context of Product operational support (eg, need for preparation for transportation in the case of Product deployment).

As opposed to a rectifying task, a supporting task cannot resolve an event. However, also operational needs can cause specific activities that belong to the group of supporting tasks from a technical point of view. Typical supporting tasks are:

- test tasks
- gain access/undo access tasks
- remove and install tasks
- assemble and disassemble tasks

5.2 A simple rectifying task

Fig 3 provides an example of a repair task of the equipment (control computer with part identifier A-001, refer to Fig 2). The example illustrates subtasks performed one after the other, without any reference to already existing supporting tasks.

The examples in this chapter provide a good understanding of task organization. This chapter should stimulate the use of different approaches to organize tasks. There are different methods of focusing on the pure activity and to separate, for example, preparation and close-up subtasks, or to include activities into a rectifying task (eg, fault location, gain access, remove, install, undo access, and test after reinstallation). It depends on the methodology determined in the LSA GD for the specific project. S3000L does not mandate one specific methodology as the only possible one.

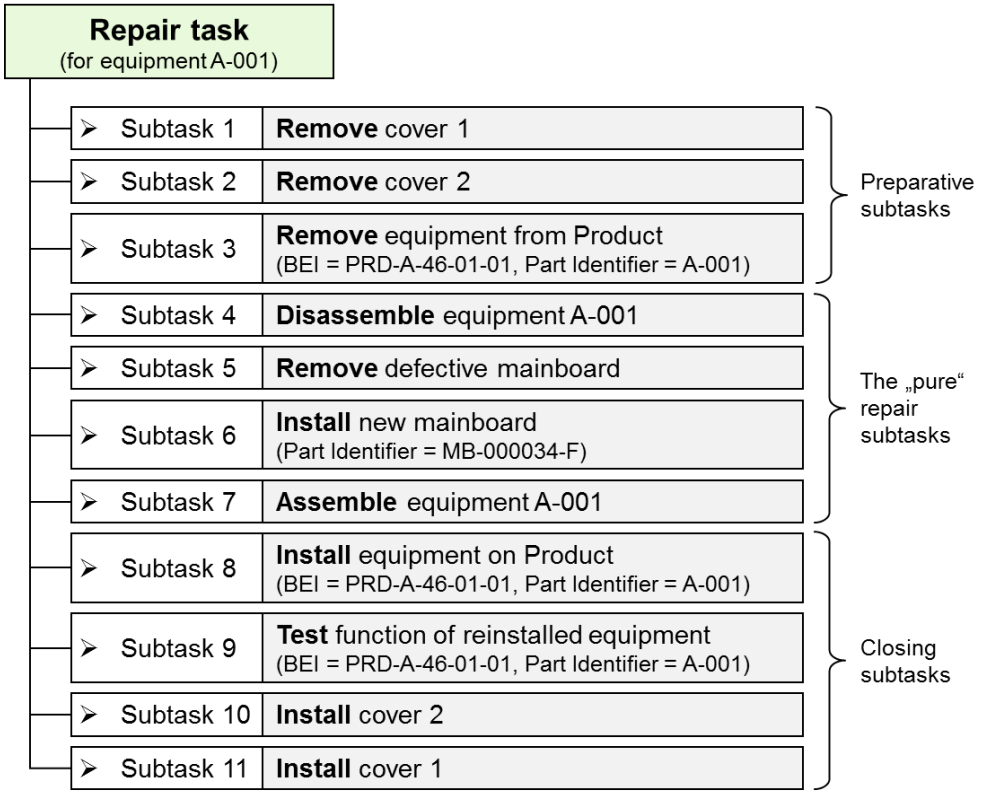


Fig 3 Simple rectifying task without references

Note

Product operation does not restart until the repair is complete.

Note

It is important to determine the LSA candidate relevant to the task. For the entire repair, the relevant LSA candidate is the equipment as a part (independent from the installation location before the removal). For the removal and installation of the equipment A-001 (subtasks 3 and 8), the installation location is the relevant information. In case of multiple installations of the equipment A-001 within the Product, the essential information is the installation location, as well as the installed part. In fact, it can be possible to install different parts from different manufacturers at the same installation location. Precondition is that the parts have the same form, fit, function, and interface. However, removal or installation tasks can be slightly different if there are different realizations. In this case, it is recommended to introduce independent tasks for the installation location and determine the applicability for the different methods for the removal or installation of different parts.

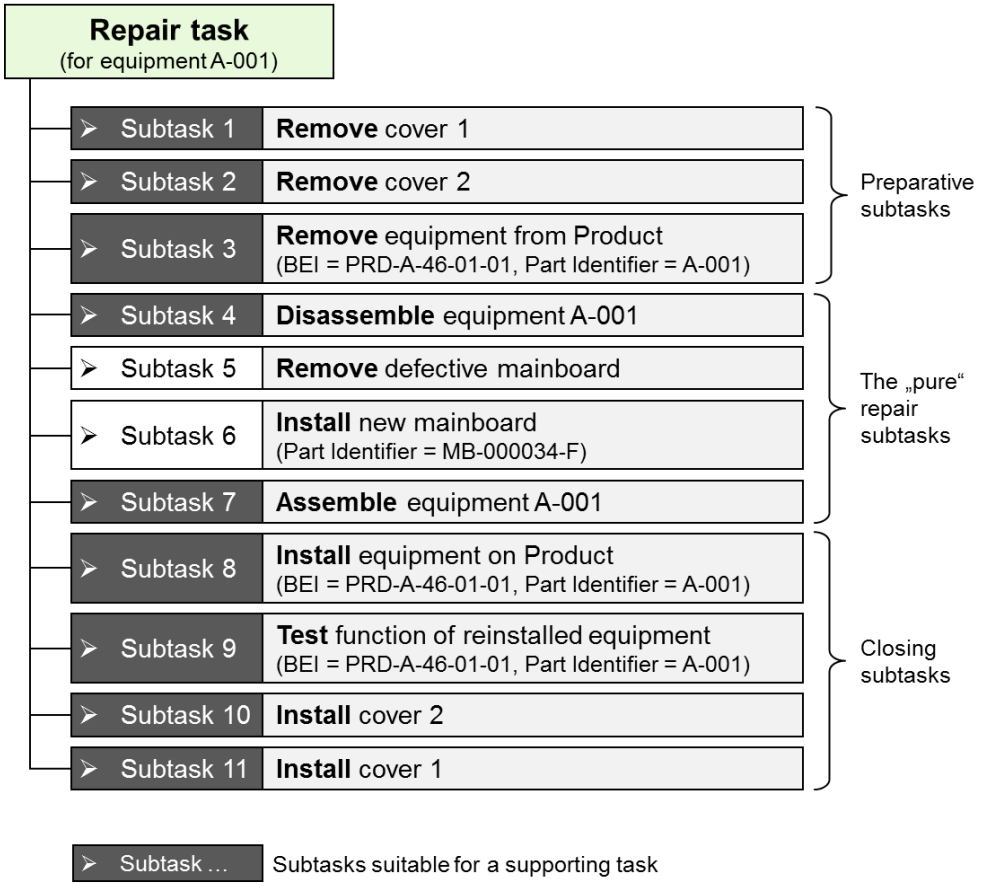
The example of Fig 3 contains subtasks that can consist of several steps, for example the removal of the equipment A-001. Additionally, there are subtasks not directly associated to the equipment A-001, but to other potential LSA candidates (eg, cover 1 or cover 2), which can be represented by a breakdown element or a part.

All subtasks within the rectifying task above, which can be subdivided into further steps and/or associated to another LSA candidate as the actual IUA (here the equipment A-001), are typical task portions of a rectifying task. They can be moved to a supporting task and later referenced to from the rectifying task.

There are some criteria to identify subtasks that can be moved to a supporting task. The most important aspect is whether other rectifying or operational support tasks can potentially reuse the supporting task, too. A typical example is a cover that must be removed to gain access to a

set of different equipment behind it. In this case, this removal task is a typical candidate to create a supporting task for the removal of the cover against the cover as the corresponding LSA candidate. Task portions associated to the IUA can also be candidates to be moved to a supporting task. For example, removal of the equipment A-001 can be reused in several different use cases (eg, in a repair or replace task of the equipment itself), but also in the context of another equipment which needs the equipment A-001 to be removed to gain access.

Taking into consideration the aspects above, a set of subtasks from Fig 3 can be identified as task components, which can be potentially moved to a supporting task, as illustrated in Fig 4.



ICN-B6865-S3000L0145-001-01

Fig 4 Task components suitable for creating a supporting task

Note

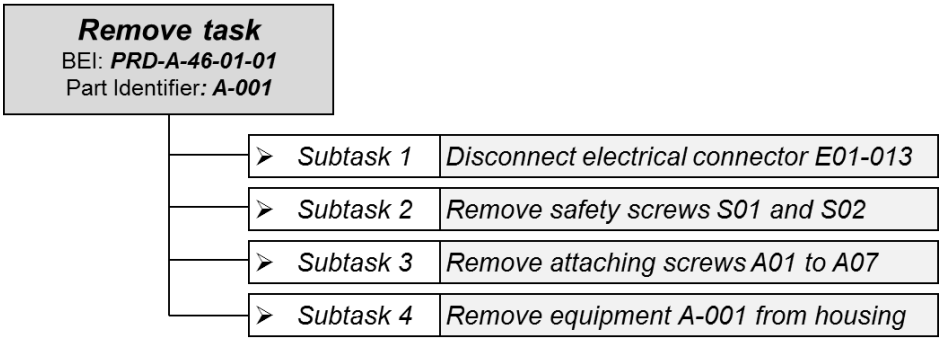
Subtasks 5 and 6 are not suitable for creating a supporting task. The removal of the defective mainboard occurs only once in the context of the analyzed repair task. This subtask will not occur in any other context. The MTA philosophy of S3000L defines this kind of subtasks as "subtask by definition", whereas referenced supporting tasks are "subtask by reference".

5.3 Supporting tasks

With the help of supporting tasks, the analyst can create a "library of task portions" that, if necessary, helps supporting the creation of more extended rectifying or operational support tasks. It is also possible to divide supporting tasks into subtasks.

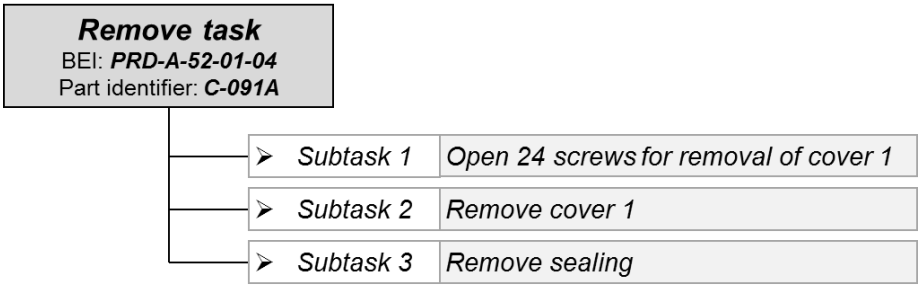
Fig 5 shows an example of the organization of the remove task for equipment A-001 in its installation location, with the indication of the corresponding Breakdown Element Identifier (BEI).

Fig 6 shows an example of the organization of the remove task for cover 1. This task does not relate to the IUA, but to cover 1 and its installation location (eg, BEI: PRD-A-52-01-04, and the part identifier for cover 1: C-091A).



ICN-B6865-S3000L0051-003-01

Fig 5 Example of a supporting task associated with the IUA itself



ICN-B6865-S3000L0052-002-01

Fig 6 Example of a supporting task associated with an LSA candidate different from IUA

Note

The tasks shown in Fig 5 and Fig 6 can equate to a procedural data module. Refer to S1000D.

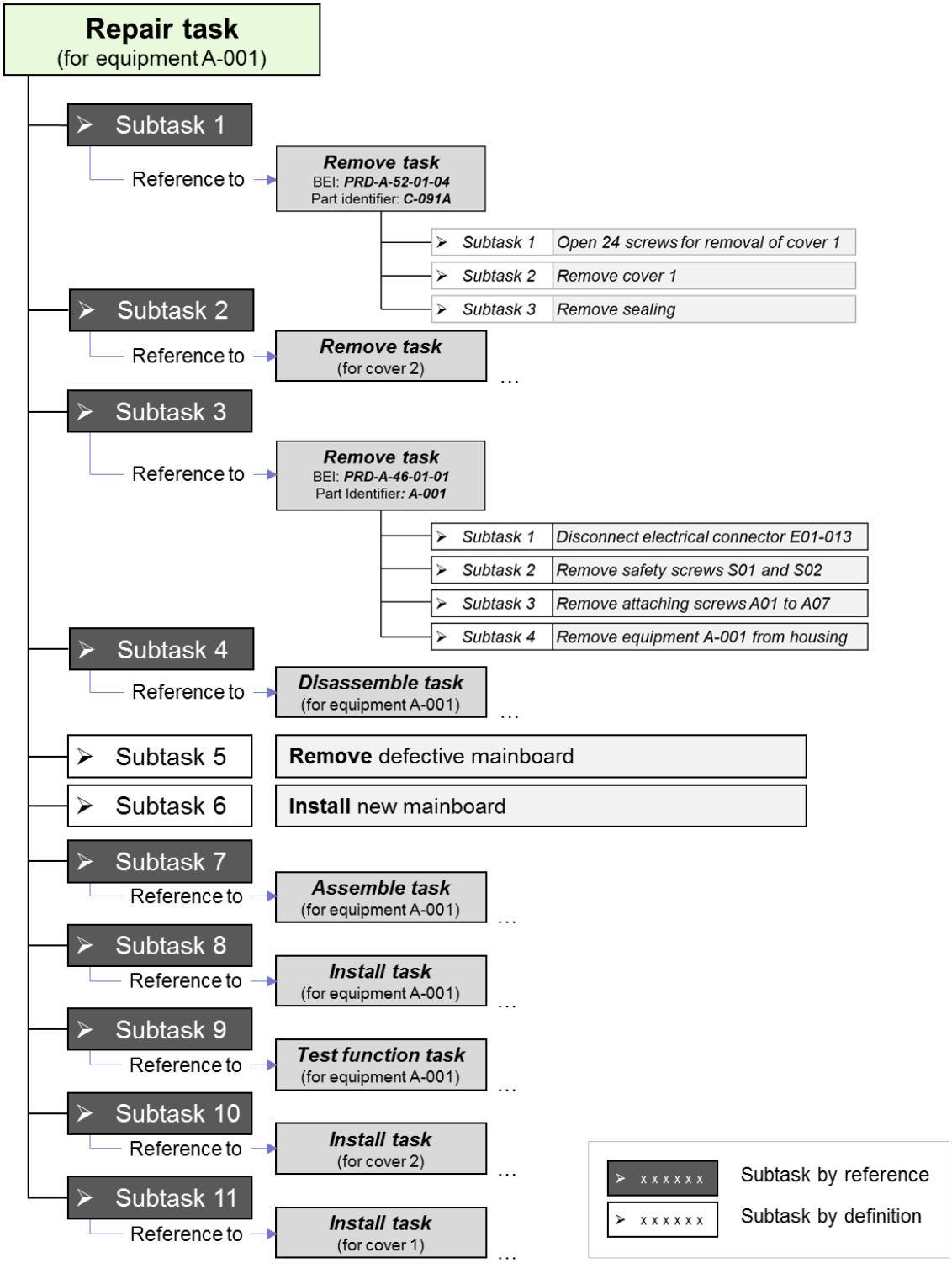
5.4 Structure of a rectifying task - referencing method

After the creation of a library of supporting tasks, the next step is to organize the rectifying or operational support tasks, accordingly. Fig 7 illustrates the use of subtasks by definition and subtasks by reference within a rectifying task. In this example, the rectifying task relates to the repair of the equipment A-001 from Fig 3, which includes subtasks by reference and subtasks by definition. There are several subtasks within this repair procedure. Other LSA candidates already describe some of these subtasks (eg, the removal of cover 1, refer to Fig 6). Some of them are already described within the same LSA candidate (eg, the removal of the equipment itself, refer to Fig 5). Using references avoids the repetition of the detailed description of single steps within the rectifying task.

Note

As a general recommendation, do not use copies of existing task structures within the LSA data.

Different tasks can reference the supporting tasks many times within a project. Therefore, for updating reasons, the LSA data must document the original information only once. In case of modifications of the supporting task, it is necessary to produce the changes only once.



ICN-B6865-S3000L0053-003-01

Fig 7 Typical rectifying task - repair procedure

Fig 7 shows a typical arrangement of a rectifying task. There are both subtasks by definition and subtasks by reference within the sequence of the entire repair task. In an extreme case, a rectifying task can contain only references to existing supporting tasks.

Note

It is generally recommended to use references for all activities that do not change, regardless the location and frequency of installation of the IUA on the Product (eg, when the item is removed from the Product, disassemble or assemble tasks are usually the same). On the other hand, installation and removal tasks can be different if the installation of the item occurs in several locations within the Product. It is necessary to decide case by case whether it makes sense to create a supporting task for remove or install activities. It depends on whether it is possible to reuse these tasks (or parts of them) in other tasks, for example in case the removal of a part is required in another context.

5.5 S1000D integration considerations

If there is a need to populate S1000D procedural data modules based upon LSA data, it is necessary to consider some aspects:

A set of preconditions can need consideration in order to perform a task, and they are of special interest for technical publications. It is required to document preconditions concerning safety in the preliminary requirements section of procedural data modules. The following aspects can be included:

- general preconditions:
  This area describes the general preliminary work to be carried out in order to achieve the necessary conditions to start the actual task.
- safety preconditions:
  This area covers all aspects necessary to achieve conditions to carry out the task in a safe environment (warnings and cautions).
- personnel preconditions:
  This area covers all aspects related to personnel. This includes the requirement for additional skills or certifications to perform the task, as well as the need for specific personnel requirements to comply with safety regulations (eg, the need for personal protective equipment to perform the task).

The sequence of the complete maintenance task (supporting or rectifying) must include as corresponding subtasks all activities necessary to reach the required preconditions. It is also possible to classify those subtasks by assigning them a specific subtask role (eg, by a role precondition). Refer to Chap 22.

Another aspect is the identification of startup and close-up activities. Within extended repair or replace procedures containing several subtasks, the preparation of equipment repair or replacement (eg, gain access or fault location procedures) can require several subtasks. The same applies to close-up activities, which do not belong to the actual repair or replace task, but must be performed to complete the maintenance task (eg, cleaning of support equipment, paperwork). It is also possible to classify those subtasks by assigning them a specific subtask role (eg, the startup or close-up role). Refer to Chap 22.

5.6 Narrative description

The narrative description of a maintenance or operational procedure is primarily a matter of technical publications. For this reason, it is mandatory to avoid performing the same activity twice. An extended description of a task within the LSA is not desirable. It is possible to provide a short description of the subtasks, using a brief wording. The LSA must identify the required support activities and analyze the identified tasks. The use of narrative texts from the LSA data as a draft for technical publications must undergo verification. Technical publications must usually observe strict rules concerning language, formatting and layout.

6 Task frequency

The expected frequency of the analyzed task is an important piece of information concerning tasks. Knowing how frequently a task must be performed each year is crucial for planning support resources. The task frequency often relates directly to the frequency of the triggering event. It is possible to foresee the frequency of some events, while some other events need to use statistic methods to estimate their frequency.

6.1 Maintenance tasks due to inherent equipment failures

Equipment failures statistically occur based on a KPI called the Mean Time Between Failure (MTBF). This value indicates the average elapsed time between two failures of an equipment. It is possible to use a simplified formula to predict the frequency of a rectifying task, assuming a constant MTBF over the entire usage time and a single installation of the equipment:

$$TF_{rec} = \sum_{i=1}^{k} FMDR_i \cdot \frac{AOR}{MTBF} \cdot \lambda_{corr} \cdot \lambda_{MB}$$

Table 8 Formula symbols (task frequency of rectifying tasks based on failures)

Formula symbol	Explanation
TF_{rec}	Frequency of the rectifying task (1/year)
AOR	Annual Operating Requirements (AOR) For equipment in continuous operation, AOR is 8760 hours per year. AOR can also have a different measurement basis (eg, driven distance in kilometers or number of cycles instead of operating hours).
$MTBF$	Mean Time Between Failure Typically, operating hours are the measurement basis of the MTBF, but there can also be a different (eg, driven distance in kilometers or number of cycles instead of operating hours).
$FMDR$	Failure Mode Distribution Ratio Several parts inside an equipment can cause the failure of the equipment. Each part/component within the equipment can be responsible for the failure of the whole equipment, with a certain ratio.
λ_{corr}	Correction factor for MTBF (in case of special conditions at special installation areas). This correction factor can differ at each installation location of equipment, and can be especially relevant in the case of a multiple installation of equipment.
λ_{MB}	Correlation factor in case AOR and MTBF have different measurement bases (eg, in case of a vehicle, the measurement basis for MTBF is kilometers instead of operational hours).
i	Index for the identification of the FMDR of the single failure mode within the analyzed equipment.
k	Number of different failure modes that are likely to accumulate.

The formula above can be considered a simplified option. It is possible to add the correction factor λ_{corr} to consider the need for correction (eg, due to environmental conditions which influence the reliability of the IUA). It is possible to use correlation factor λ_{MB} to consider

different measurement bases for AOR and MTBF. In case corrections or correlations are not necessary, both factors equal 1.

Note

Within the LSA GC, the customer and contractor must harmonize and determine any correction or correlation factor to be used in a project.

A more complex formula that takes into account other influence factors allows calculating task frequency more precisely. The influence factors include:

- no failure found factors (eg, a built-in test cannot duplicate the failure)
- if the MTBF is not constant, but is a function of time MTBF(t), it is necessary to use integral calculation
- different types of MTBF (eg, predicted, allocated, measured) to be selected
- correction required because of induced failures

The MTBF is particularly interesting when analyzing different types of equipment. The distribution of the MTBF value over the entire life cycle of equipment can have a different behavior depending on the type of equipment. MTBF values can change over the life cycle of equipment. For example, a mechanical equipment with deterioration effects shows a different behavior than an electronic equipment. Reliability analysis activities will perform such examinations. Mathematical functions help describing in detail different failure distribution behaviors as a function of time. In this context, refer to special literature concerning reliability.

In case of multiple installation of equipment, it is recommended to calculate the task frequencies for each installation location separately. It is possible to summarize the calculated single task frequencies for a required accumulation of task frequencies (eg, for maintenance tasks being performed in a special repair shop). Depending on the installation area, the frequency of the failure of equipment can be significantly different.

6.2 Task frequency for supporting tasks

It is not possible to calculate a task frequency for supporting tasks the same way as for rectifying tasks. Usually, there is not a direct link between supporting tasks and any event like rectifying tasks. However, rectifying tasks normally call supporting tasks by using references. By adding the task frequencies of the referencing rectifying tasks, it is possible to use the simplified formula below to calculate an accumulated task frequency of any supporting task:

$$TF_{supp} = \sum_{i=1}^{n} \lambda_Q \cdot TF_{rec,i}$$

Table 9 Formula symbols (task frequency of supporting tasks)

Formula symbol	Explanation
TF_{supp}	Task frequency of the supporting task (1/year)
$TF_{rec,i}$	Task frequency of the referencing rectifying task (1/year)
λ_Q	Number of "calls" of the rectifying task for the supporting task to be calculated
i	Index for the referencing rectifying tasks
n	Number of the referencing rectifying task

Since different rectifying or operational support tasks can reference a supporting task, the calculation of a task frequency for the supporting tasks can be interesting for statistical figures used to:

- identify maintenance drivers to be assessed for potential improvement of design
- quantify maintenance tasks depending on the actual use/maintenance of an item

Example: Opening and closing of covers

Different rectifying tasks from many different LSA candidates can reference several times the remove and install tasks of a cover. It can be important to know the frequency of installation/removal of a cover within one year, because of the potential need for some spare parts after a certain number of remove and install tasks. Inherent failures or actual damage that require the removal or installation of the cover usually constitute only a small fraction of the actual removal and installation activity. Many cases result from the need to gain access to other equipment behind the cover.

Because of a frequent removal and installation of a cover, the design must be user friendly (eg, use of quick release fasteners) and damage tolerant.

Note

Different types of events, which potentially can have impact among themselves, can trigger the referencing tasks. For example, a failure or a PMTRI can trigger the replacement of an item. As a consequence, a corrective replacement (with the interval being reset after corrective replacement) influences the task frequency of the preventive replacement. Refer to Para 6.5.

6.3 Maintenance tasks due to damages or special events

It is not possible to predict equipment failures for other reasons in the same way as described in Para 6.1. In fact, it is only possible to make an estimate of the task frequency. If there is any experience concerning the occurrence of damages or special events, for example with the help of statistical investigations, the results provide an estimate of the frequency of these unpredictable events and the corresponding maintenance tasks.

However, it is difficult to forecast the frequency of these events, it is important to get an estimate based on the best information available. All maintenance tasks triggered by these events have their corresponding requirements concerning material and personnel. It is necessary to take into account these unpredictable events to estimate the resources and the capacity utilization.

6.4 Preventive maintenance tasks

In the case of preventive maintenance tasks with repeated time limits (often referred to as interval), no additional calculation or estimation is required, because the corresponding PMTRI and a later packaging of preventive maintenance tasks (refer to Chap 10) determine the interval of the maintenance activity.

It is possible to convert repeated time limits to task frequency values based on how often they occur per year. It is necessary to relate each interval determined with a specific measurement basis value to the annual operating requirements. The following simple formula calculates the task frequency:

$$TF_{sched} = \frac{AOR}{TL_{rep}}$$

Table 10 Formula symbols (task frequency scheduled maintenance tasks)

Formula symbol	Explanation
TF_{sched}	Task frequency of a scheduled task (1/year)
AOR	Annual operating requirements For equipment in continuous operation, AOR is 8760 hours per year. AOR can also have a different measurement basis (eg, distances or number of cycles instead of operating hours).
TL_{rep}	Repeated time limit (same measurement base as AOR)

Note

Preventive maintenance can use a more complex structure to determine different thresholds, intervals and triggers for a scheduled maintenance task (refer to Chap 10). In this case, it is often impossible to calculate a simple task frequency value.

6.5 Task frequency of replacement tasks

A specific situation concerning task frequency can occur if different types of events can trigger a task. A typical example for this situation is the replacement of an equipment, as there can be various triggers, including:

- replacement of equipment with a new one due to a failure or a damage (and due to the required high availability of the complete Product, which means the Product cannot wait for an equipment repair)
- replacement of the equipment because of a limitation by authorized life
- replacement of the equipment after a certain time limit due to a PMTRI

In this case, a simple mathematical formula is not suitable to calculate the final task frequency, because the different characteristics of the replacements can have impact among themselves. For example, an unscheduled replacement due to an equipment failure will have an impact on the preventive maintenance task with a fixed interval, because it is necessary to reset the determined interval.

7 Task resources

7.1 General aspects

The identification of task resources for all identified tasks is crucial in defining the maintenance solution and supporting the planning for a timely provisioning of all task resources. Task resources can be:

- personnel
- material
  - spare parts
  - consumables
  - raw material
  - support equipment (eg, special tools, handling equipment, test equipment, storage or transport containers, standard hand tools)
- facility/infrastructure
- documentation

During the development of a Product, material task resources such as support equipment and facility/infrastructure, are often developed in parallel. In this case, the MTA can identify the need for a resource, which normally leads to the creation of a specification document including the requirements for the identified resource. LSA data can document this to indicate the identified task resources, and confirm the provisioning process has started by creating a task resource

specification (material resource by specification, refer to Chap 19). After the completion of the development and/or provisioning process for a specified task resource, the concrete material or facility/infrastructure resource can be documented in the LSA data and it is possible to relate it to the corresponding resource specification.

7.2 Task resource assignment

It is necessary to assign the task resources at the relevant level within the task itself. Generally, a user of a maintenance documentation must be able to identify the need for a resource within the sequence of the task. With respect to training, it is interesting to determine which personnel needs appropriate competence to use and/or operate a special support equipment. The principle of the structure of tasks (refer to Para 5), which can be organized by subtasks including references to supporting tasks, determines resources allocation. It is recommended to follow the basic principle that each resource is linked to the activity that requires that resource.

In general, the S3000L data model allows analysts to assign task resources to the complete task or to a subtask within a task. Refer to Chap 19. However, it is required to determine the consequences of a resource assigned to the entire task.

- Is the resource valid for all subtasks (including subtasks by reference) within the task (eg, assigning personnel resources at task level)?
- Is the resource the result of an aggregation and consolidation of resources coming from subtasks (by definition and by reference)?

For a clear understanding, it is recommended to establish appropriate business rules, for example within the LSA GD.

Table 11 gives an overview of potentially typical assignment methods for task resources.

Table 11 Assignment of task resources

Task resource	Assignment aspects
Personnel task resources	Personnel is assigned to the subtasks that require it. If the complete task requires the same personnel, it is possible to assign it to the task instead of each single subtask (as determined by specific business rules).
Material task resources - spare parts and consumables	Spare parts and consumables are typically assigned to subtasks, where they are required. As an alternative, an aggregation and consolidation approach allows spare parts and consumables to be assigned to the complete task. Specific business rules must determine this approach.
Material task resources - raw material	Raw material is typically assigned to subtasks, where it is required. Often the raw material can be assigned to those subtasks, where the preparation is performed before finally install the modified raw component into the Product. As an alternative, an aggregation and consolidation approach allows raw material to be assigned to the complete task. Specific business rules must determine this approach.

Task resource	Assignment aspects
Material task resources - Support equipment	Support equipment is typically assigned to the subtask that requires it. If the complete task requires support equipment, it is possible to assign it to the task instead of each single subtask. Specific business rules must determine this approach. **Note** Support equipment indicates tools or test equipment. However, it is necessary to consider also support equipment for operational support like transport or storage containers, fixation belts or ground-handling devices for an aircraft (eg, a towing bar for aircraft). It is also possible to add information about which personnel uses or operates which support equipment. Within complex subtasks, different people can use different pieces of support equipment. For training requirements, it must be possible to associate support equipment with the relevant personnel resource.
Facilities and infrastructure task resources	Facilities and infrastructure are typically assigned to a complete task.
Document task resources	A document task resource identifies a document (including digital documents) to be filled out in the context of a close-up paperwork. **Note** Typical task resources usually do not include the required technical publications (eg, maintenance manuals) that support the performance of a task. The document assignment mechanism must assign those documents to the relevant task/subtask. Refer to Chap 19.

It is necessary to summarize and harmonize all resources from the subtasks and referenced supporting tasks at complete task level. LSA data evaluation or reports concerning the different resources (or even for all resources in one report) can provide this summary. Fig 8 provides an example.

Install task for equipment A-001

Subtask	Spare parts	Consumables	Support equipment	Personnel [role]	Personnel [competence]	Personnel [labor time]
Install equipment A-001 into housing	Seal [1x]	Adhesive C [as required]	None	Performer Helper	Electrician (basic) Electrician (basic)	1 min 1 min
Install attaching screws A01 to A07	Washer [7x]	None	Phillips screwdriver, size 2 [1x]	Performer	Electrician (basic)	3 min
Install safety screws S01 and S02	None	None	Torque wrench [1x]	Performer	Electrician (basic)	2 min
Connect electrical connector E01-013	None	None	None	Performer	Electrician (basic)	0,5 min

ICN-B6865-S3000L0054-003-01

Fig 8 Example of task resources assignment at subtask level

Table 12 summarizes the resources of the install procedure from Fig 8 at task level.

Table 12 Aggregation of task resources

Resource type	Resource	Quantity
Spare parts	Seal	1
	Washer	7
Consumables	Adhesive C	as required
Support equipment	Torque wrench	1
	Phillips screwdriver, size 2	1
Personnel	Electrician (basic)	2

Note

It is necessary to distinguish each individual performing a task from one another. The simplest differentiation is just indicating the number of persons, for example: person A, person B, person C, etc. Additionally, it is possible to specify the role of a person within a task, for example the performer, the helper, the supervisor, the auditor. These two indications can be used together, for example: performer A, performer B, etc. The corresponding business rules must address the use of role classification. Additionally, the value of personnel labor time can exceed the task duration if two or more people work simultaneously on the same subtask (refer to the first subtask in Fig 8).

7.3 Resources out of references

When using the methodology of referencing supporting tasks, it is necessary to take into account that the referencing task will include the resources of the referenced tasks. At first glance, this seems to be an appropriate approach. However, this approach can result in a false estimate of personnel requirements. Personnel with a lower competence usually performs simple subtasks (eg, the removal of an access panel). Therefore, simple supporting task usually indicate this type of personnel. In case a person with a higher qualification performs an extended repair or replace task that references the supporting task, it is necessary to consider that it will be very likely for the person with a higher qualification to perform the referenced supporting task as well. The corresponding business rules must determine how to handle this situation and how to document a potential change of personnel within a referenced supporting task. It is also possible to use the definition of competences, which include other competences.

8 Task location aspects

It is possible to distinguish different location information types to cover all aspects concerning the location where task performance occurs.

8.1 Location in conjunction with maintenance level

The most general information is the determination of a maintenance level. This indicates whether a task performance occurs on customer site or on industry site. It is necessary to define at an early stage the number of maintenance levels valid for the project (especially on customer side), and this information will be later used within a LORA. Refer to Chap 11.

8.2 Location in conjunction with the Product itself

It is possible to align the location of a task in conjunction with the Product with the S1000D data element Item Location Code (ILC). The terms on- and off- tasks indicate the location of the maintenance activity with regard to the Product. An on-task concerning location means that the activity is performed directly on the Product (eg, in or directly at an aircraft). A removal task of an LRU is always an on-task, because the LRU is always removed directly from the Product. A

typical example for an off-task concerning this type of location is the disassemble task after the removal in a special maintenance shop.

8.3 Location in conjunction with the required facility

The attribute of a required facility is another piece of information concerning location of a task. In addition to the maintenance level from Para 8.1 and the on- and off- information from Para 8.2, the required facility is another typical attribute assigned at task level. The selection of a facility object as the location of a subtask does not automatically fix the location information concerning on- and off-activities. Examples for facilities are:

- individual workshops such as electronic shop or engine shop
- maintenance hangar
- dockyard
- clean room

8.4 Location in conjunction with a Product zone

If the Product is divided into physical areas (zones), this information serves as additional information on the location of task performance. In this case, the location aspect relates to the layout of the Product. The information helps estimating the amount of activity within each zone of the Product. This can influence design, for example by making it easier to access a zone where maintenance tasks frequently occur.

Note

The use of information concerning zones helps documenting the result of a zonal analysis in conjunction with preventive maintenance. In this case, it is possible to document zones using non-hardware breakdown elements. The identified PMTRI and the corresponding tasks can be associated with the zonal breakdown elements (refer to Chap 4 and Chap 10).

9 Additional aspects for tasks

9.1 Product availability during maintenance performance

The impact of maintenance tasks on the operability of the Product is another important piece of information concerning the Product and different systems within the Product. Examples of impacts include, but are not limited to:

- Product/system inoperable during equipment maintenance
  In this case, the Product/system cannot be used during, for example, a repair procedure or an inspection. The Product/system must wait until the completion of the maintenance task before it is available again. A typical situation can be the removal of a defective component and the repair of the defective component (eg, in a special maintenance shop). After the successful repair, the repaired component will be reinstalled into the Product. During the entire repair activity on the defective component, the Product/system operation is on hold.

- Product/system operable during equipment maintenance with reduced capability
  In this case, it is possible to use the Product/system during, for example, a repair procedure or an inspection with reduced capability (eg, a specific subsystem is not available during maintenance, but the rest of Product is fully operational).

- Product/system operable during equipment maintenance
  In this case, it is possible to use the Product/system with full capability during a repair procedure or an inspection. A typical example can be a visual inspection task of some equipment during full operation. For the inspection, there is no need to interrupt the operation or the mission.

Note

The data element *taskOperabilityImpact* in Chap 22 provides concrete examples for classification values to describe Product availability during maintenance tasks.

9.2 Tasks and applicability

Maintenance or operational support tasks can vary depending upon the conditions during task performance. The task itself can be nearly the same but require different or additional support equipment, or it can be slightly different from the original one. In this case, it is recommended to create different tasks to document the different "solutions" for the different conditions in which task performance occurs. It is possible to assign the different solutions to specific applicability statements. The following list gives a set of examples for the creation of corresponding applicability statements for different solutions for specific tasks:

- performance of the task under different environmental conditions (eg, under tropical conditions or under arctic conditions)
- performance of the task at a different location (eg, out of the area or in a workshop)
- performance of the task for another customer with different preconditions concerning personnel and equipment available at the operational site
- performance of a repair task to provide a permanent repair or a temporary repair
- performance of a task under peace time or war time scenarios

Note

LSA data can include applicability statements as described above, which can also be forwarded to technical publications. Refer to S1000D.

9.3 Task duration

Information on the duration of the task provides an overview about the capacity utilization of personnel and material resources. Within the task, it is necessary to separate the duration of a task and the working time of the personnel (also called labor time). Table 13 explains the difference between task duration and labor time.

Table 13 Task duration and labor time

Time	Description
Task duration	Duration of the entire task. The duration documentation can relate to either the entire task or the single subtasks. It is possible to calculate the duration of the entire task as a proper accumulation of durations coming from the subtasks including referenced supporting tasks and taking into account parallel activities. Refer to Para 9.4. **Note** The term Mean Elapsed Time (MET) is also commonly used for task duration.
Labor time	Accumulated time of personnel work. It is possible to document this time in relation to either the entire task or the single subtasks. If more than one person is working at the same time on a task/subtask, it is necessary to sum the labor time of all persons, taking into account the different competences of each individual. For example, three people with the same level of competence are working in parallel on a subtask, which takes 20 minutes. In this case, the documentation includes the following data: Subtask duration: 20 minutes Labor time: 60 minutes

Regarding times/durations of tasks, it is necessary to consider the type of documented duration. The time doing the job is often the only documented type of duration. This means that all non-productive activities (eg, paperwork) are usually not included. To take into consideration non-productive times, it is possible to make an estimate in the LSA data.

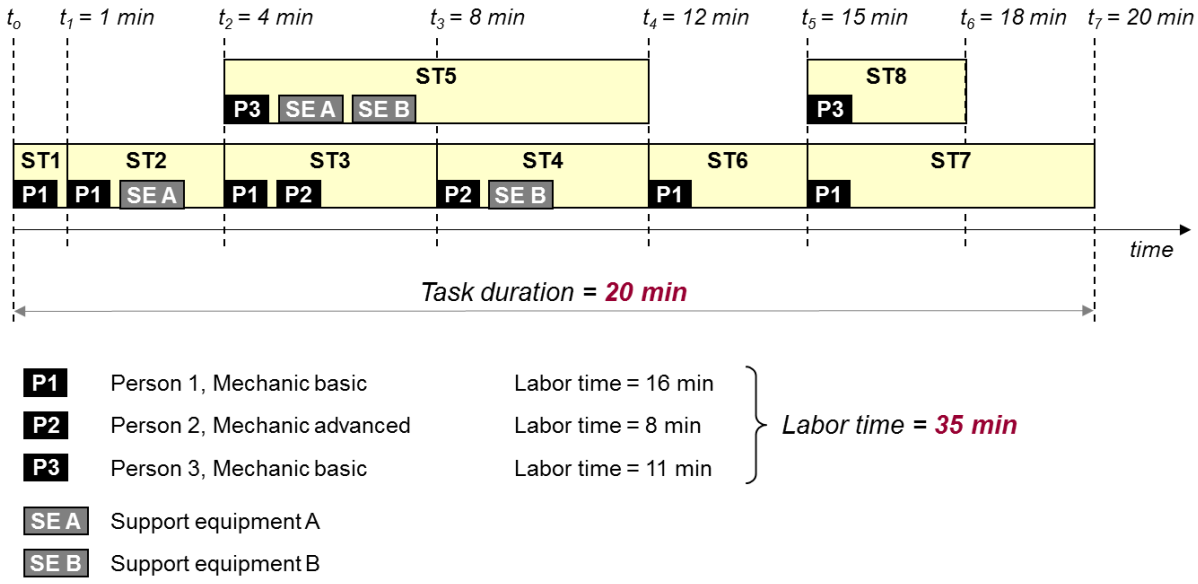
Another aspect is the consideration of logistics delay times. LSA usually does not document any delay caused by waiting times for support resources (eg, support equipment or facilities not immediately available, waiting for personnel, waiting for material). To take into consideration logistics delay times, it is possible to make an estimate outside the LSA.

Note

Task duration must include the waiting times caused by integral parts of a maintenance task (eg, the curing of material or the drying of paint), which are therefore not considered the same way as non-productive times or logistics delay times. The same holds for an authorized inspection which is mandatory before moving to the next step of the complete task. These easily predictable task portions can be documented within a task as a subtask with its specific duration and support resources.

9.4 Parallel activities within tasks

It is necessary to consider the simultaneous performance of subtasks in more complex maintenance or operational support tasks that contain many subtasks performed by different people. Activities occurring simultaneously influence the duration of the complete task, as well as the task resources concerning support equipment and personnel. Refer to the example in Fig 9:



ICN-B6865-S3000L0055-002-01

Fig 9 Parallel activities and their resources/durations

Subtask **ST5** is simultaneous to **ST3/ST4** and requires the support equipment **SE A** and **SE B**. For subtask **ST5,** it is possible to use the same support equipment as in subtask **ST2,** because these activities are not performed in parallel. The consequence for the complete task is the requirement of one piece of support equipment **SE A**. The situation is different for support equipment **SE B**. Subtasks **ST4** and **ST5** occur simultaneously and both require **SE B.** This means two pieces of support equipment **SE B** are needed for the complete maintenance task.

The situation is similar for personnel as well. The person with the basic mechanic qualification is required for subtasks **ST3** and **ST5,** occurring simultaneously, as well as for **ST7** and **ST8**. This requires an additional person **P3** with the same qualification as person **P1**.

Table 14 shows the differences between a simple approach using a sequential methodology (ie, one subtask after the other) or the possibility of simultaneous activities.

Table 14 Comparison of resources for sequential or parallel subtasks

Step by step sequence		Parallel activities	
Task duration: 35 min		Task duration: 20 min	
Support equipment requirements:			
SE A: 1		SE A: 1	
SE B: 1		SE B: 2	
Personnel requirements:			
Mechanic basic:	1 (P1)	Mechanic basic:	2 (P1 and P3)
Mechanic advanced:	1 (P2)	Mechanic advanced:	1 (P2)

The S3000L data model allows documenting both the parallel and sequential performance of subtasks. Refer to Chap 19. In addition to the simple attribute of a subtask duration, it is possible to use a timeline construct to document parallel activities, with a clear indication of the start and the end of each subtask. Additionally, a timeline leg is defined, for example, for waiting times before the dependent subtask can start. In this context, LSA data evaluation can help create a draft version of a working plan, for example by obtaining a graphic schedule of a maintenance task directly from the LSA data.

9.5 Training requirements

Information on the level of competence of personnel performing any task documented in the LSA data can provide the basis for the identification of training requirements. In this context, it must be distinguished between special training needs and training needs covered by normal professional education. However, especially in many international projects, education and potential certification in each nation can make it difficult to identify general training needs valid for each individual customer and/or Product operator.

To support TNA there is a need to document special training requirements to perform a maintenance or operational support task. For this reason, the task analysis must consider the requirements at subtask level (eg, the use of a complex support equipment that requires training), as well as at task level, to assess the complexity of the task as a whole. The aspects under consideration are:

- the section or department in charge of the task
- the selected skill level of personnel performing the subtask
- the need for team training due to the complexity of the task
- the use of special support equipment that requires training
- required special training to perform the task/subtask
- required training methods
- required experience to perform the task/subtask
- the complexity of the task as a whole

The information concerning training can be collected at task or subtask level. At subtask level, it is possible to analyze each activity in detail and to assign special support equipment to the person performing the activity. Training information collected at task level is valid for all subtasks (eg, determination of a general competence required for the task).

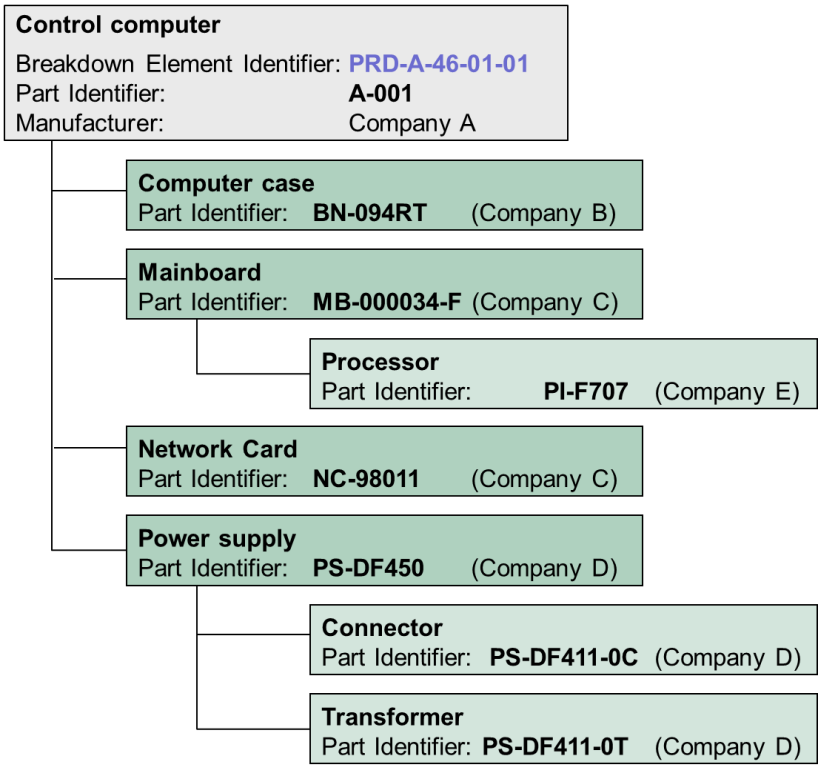
9.6 Circuit breaker settings

The correct setting of circuit breakers is often an important part of a subtask, or even a subtask in itself. It can be necessary to set several circuit breakers in a specific sequence. In the S3000L data model (refer to Chap 19), it is possible to assign the circuit breakers and the corresponding settings (including the sequence) to a concrete subtask. This helps in

determining the important preconditions concerning circuit breakers to perform the subtask in a safe condition.

10 Examples for different maintenance solutions

The following examples provide a better understanding on how to assign tasks to the corresponding LSA candidates and how to document the different maintenance solutions. The equipment (control computer) which is used for the examples is illustrated in Fig 10. The complete control computer is installed in a land vehicle and is represented by a breakdown element within the Product breakdown of the vehicle. The realization of the breakdown element (BEI = PRD-A-46-01-01) is done by the installation of the control computer with a part identifier A-001.



ICN-B6865-S3000L0050-002-01

Fig 10 General equipment example

The following examples also provide a better understanding about how different frame conditions or requirements given by operational needs of the operator can influence maintenance tasks. The examples illustrate the diversity of maintenance solutions and the corresponding tasks. They go from a rather simple situation like an equipment replacement to an extended situation with several activities at several maintenance levels.

10.1 Example 1: Major failure, computer case broken

Event:

Complete malfunction of the control computer due to a broken computer case after a heavy shock. No repair possible, the complete control computer must be replaced.

Special operational requirements:

None

LSA candidate:

Control computer (breakdown element)

Rectifying maintenance task for the event:

Replace the control computer at its installation location. Refer to Table 15.

Table 15 Subtasks for example 1

Step	Description	Maintenance level/task location
1	Remove the faulty control computer	Customer's vehicle workshop
2	Install a new control computer	Customer's vehicle workshop
3	Test the new control computer	Customer's vehicle workshop

Potential follow-on tasks after the rectifying task:

- disassemble faulty control computer to recycle reusable components
- disposal of all components which cannot be recycled

10.2 Example 2: Failure of the mainboard, case 1

Event:

Failure of the mainboard, the specific failure is not repairable by changing components on the mainboard.

Special operational requirements:

None

LSA candidate:

Control computer (part)

Rectifying maintenance task for the event:

Repair control computer by replacement of mainboard. Refer to Table 16.

Table 16 Subtasks for example 2

Step	Description	Maintenance level/task location
1	Remove the faulty control computer	Customer's vehicle workshop
2	Disassemble the faulty control computer	Customer's computer workshop
3	Remove the faulty mainboard	Customer's computer workshop
4	Install a new mainboard	Customer's computer workshop
5	Assemble the control computer	Customer's computer workshop
6	Test the control computer in the workshop	Customer's computer workshop
7	Install the repaired control computer on vehicle	Customer's vehicle workshop
8	Test the function of control computer after installation on a vehicle	Customer's vehicle workshop

Potential follow-on tasks after the rectifying task:

- disposal of the mainboard

Note

It is also possible to include only subtasks 2 to 5 in the repair task, as the other subtasks do not relate directly to the pure repair activity. All tasks related to preparation or close-up can be regarded as not being relevant for repair. In this case, it is necessary to document the preparation (eg, remove equipment) and close-up activities (eg, test in workshop, installation, further test at the installation location) as separate tasks, and link them to a corresponding task requirement. The adopted methodology will determine how to organize the different types of tasks. This is also valid for all examples in Para 10.

10.3 Example 3: Failure of the mainboard, case 2

Event:

Failure of the mainboard, the specific failure is repairable by replacing a component (eg, processor) on the mainboard. The replacement of the processor can occur at customer site.

Special operational requirements:

None

LSA candidate:

Control computer (part)

Rectifying maintenance task for the event:

Repair the control computer by repair of the faulty mainboard (ie, replace the faulty processor). Refer to Table 17.

Table 17 Subtasks for example 3

Step	Description	Maintenance level/task location
1	Remove the faulty control computer	Customer's vehicle workshop
2	Disassemble the faulty control computer	Customer's computer workshop
3	Remove the faulty mainboard	Customer's computer workshop
4	Repair the faulty mainboard by replacement of the faulty processor	Customer's computer workshop
5	Install the repaired mainboard on the control computer	Customer's computer workshop
6	Assemble the control computer	Customer's computer workshop
7	Test the control computer in workshop	Customer's computer workshop
8	Install the repaired control computer on vehicle	Customer's vehicle workshop
9	Test the function of control computer after installation on a vehicle	Customer's vehicle workshop

Potential follow-on tasks after the rectifying task:

– disposal of processor

The solution described above has an impact on the availability of the Product. The Product is not used before the completion of the repair activities. During repair time, the Product is not operable. A possible alternative solution is replacing the complete control computer and repairing the faulty unit later, when the Product is already in use again. Table 18 describes the two corrective maintenance solutions and their corresponding consequences.

Table 18 Influence of corrective maintenance options on Product's availability

Product waiting situation	Consequence
Case A: Product is waiting for repair of equipment	The entire equipment is not needed as a spare part, because the same equipment will be reinstalled, after a successful repair, into the Product Spare parts to repair the faulty equipment are required. Product availability is reduced due to the waiting time.
Case B: Product is not waiting for repair of equipment	The entire equipment is needed as a spare part because a new equipment will be installed to repair the Product and allow reusing the Product as soon as possible. The repair of the equipment will occur later when the Product is already in use. Spare parts are necessary to repair the faulty equipment. Availability of the Product is better than the availability in case A. There is no extended waiting time for the completion of the equipment repair.

10.4 Example 4: Failure of the mainboard, case 3

Event:

The specific failure of the mainboard is repairable by replacing a component (processor). The replacement of the processor can only occur at industry level. The customer does not want to wait until the mainboard is repaired at industry level.

Special operational requirements:

The vehicle must be available again as soon as possible. No acceptable waiting time for the control computer repair.

LSA candidate:

Control computer (breakdown element)

Rectifying maintenance task for the event:

Replace the control computer. Refer to Table 19.

Table 19 Subtasks for example 4

Step	Description	Maintenance level/task location
1	Remove the faulty control computer	Customer's vehicle workshop
2	Install a new control computer	Customer's vehicle workshop
3	Test the new control computer	Customer's vehicle workshop

Potential follow-on tasks after the rectifying task:

- repair the faulty control computer by replacement of the mainboard
  LSA candidate for the follow-on task: the control computer (part), refer to example 3 in Para 10.3
- prepare for transportation and transport the faulty mainboard to industry
  LSA candidate for the follow-on task: mainboard (part), the task will be described in the LSA as an operational support task

– repair faulty mainboard by replacement of the processor at industry (industry task ⇨ no need to include this task in the LSA data)

Note

Example 4 is more extended and shows more follow-on tasks. It is obvious that the distribution of activities influences the requirements for task resources, the means to make available spare parts, support equipment and competent personnel at the corresponding facilities where the different tasks are performed. For example, after a successful repair of the mainboard at industry, it will be sent back to the customer and will be available again as a spare part for further repair tasks. Different maintenance solutions have a significant impact on spare part life cycle.

10.5 Example 5: Complex maintenance procedure on several levels

Event:

Failure of the power supply. The faulty part is repairable at operational site by replacing the component (transformer). The replaced component is repairable at an industry site or at a special maintenance facility of the customer.

LSA candidate:

Control computer (part)

Special operational requirements:

None, Product can wait for the corrective maintenance of the equipment.

Rectifying maintenance task for the event:

Repair the control computer by replacement of power supply. Refer to Table 20.

Table 20 Subtasks for example 5

Step	Description	Maintenance level / task location
1	Remove the faulty control computer	Customer's vehicle workshop
2	Disassemble the faulty control computer	Customer's computer workshop
3	Remove the faulty power supply	Customer's computer workshop
4	Install a new power supply	Customer's computer workshop
5	Assemble the control computer	Customer's computer workshop
6	Test the control computer in workshop	Customer's computer workshop
7	Install the repaired control computer on vehicle	Customer's vehicle workshop
8	Test function of control computer within the vehicle	Customer's vehicle workshop

Potential follow-on tasks after the rectifying task:

– repair of power supply by replacement of the transformer at customer site
LSA candidate for the follow-on task: power supply (part). Task will be included in the LSA as a rectifying task.

– prepare for transportation and transport faulty transformer to industry or to specific maintenance facility of the customer (both tasks are possible, as described above).
LSA candidate for the follow-on task: transformer (part). The task will be included in the LSA as an operational support task.

- repair faulty transformer at specific maintenance facility of the customer.
  LSA candidate for the follow-on task: transformer (part). Task will be included in the LSA as a rectifying tasks.

- repair faulty transformer at industry (industry task ⇨ no need to be included in the LSA)

10.6 Summary of examples

The examples within Para 10 show a variety of maintenance solutions/scenarios. The examples are just a part of the possible scenarios and solutions. The aim of these examples is to highlight that there are, in general, different possibilities to implement maintenance solutions. Considerations on the most effective maintenance solution for the specific usage scenario for an individual customer always take part in the choice of the specific maintenance task, at a specific maintenance level or at a specific facility. Several criteria can influence this selection and can be, but are not limited to:

- the dominating aspect is the availability of the Product, need to minimize down time
- the dominating aspect is cost, maintenance must use the most cost-effective approach
- the degree of capacity utilization concerning customer-operated maintenance facilities must be high or at least on a specific level
- the need to minimize dependency from industry support (this aspect can be especially relevant for military products)
- the need to preserve on establish competence on customer side
- the maintenance strategy is pre-assigned, no foreseen repair activities at customer site
- the lack of capabilities on customer side

11 Associated parts of the S3000L data model

The documentation of the associated data to this chapter is supported by the following Units of Functionality (UoF), refer to Chap 19:

- S3000L UoF Change Information
- S3000L UoF Circuit Breaker
- S3000L UoF Competence Definition
- S3000L UoF Environment Definition
- S3000L UoF Facility
- S3000L UoF LSA Candidate
- S3000L UoF Location
- S3000L UoF Performance Parameter
- S3000L UoF Resource Specification
- S3000L UoF Task
- S3000L UoF Task Requirement
- S3000L UoF Task Resource
- S3000L UoF Task Usage
- S3000L UoF Time Limit

Chapter 13

Software support analysis

Table of contents

List of tables

List of figures

References

Table 1 References

Chap No./Document No.	Title
Chap 3	LSA process
Chap 4	Product structures and change management in LSA
Chap 10	Development of a preventive maintenance program
Chap 11	Level of repair analysis
Chap 12	Task requirements and maintenance task analysis
Chap 19	Data model
S4000P	International specification for developing and continuously improving preventive maintenance
RTCA/DO-178B	Software Considerations in Airborne Systems and Equipment Certification

Chap No./Document No.	Title
SAE JA1004	Software Supportability Program
SAE JA1005	Software Supportability Implementation Guide
SAE JA1006	Software Support Concept

1 General

1.1 Introduction

In modern products, supportability aspects for software have become increasingly important. Complex software packages support or realize functionality more and more. Concepts and processes for hardware components are established to guarantee full Product supportability during its entire life cycle. The appropriate tool to achieve this goal for hardware is the LSA process. In principle, the same requirements apply for software. Hardware and software are of equal importance for a Product's proper operation. For this reason, it is possible to apply an analysis methodology called Software Support Analysis (SSA) to develop an adequate Software Support Concept (SSC). SSA is a methodology that analyzes software with the purpose of supplying the customer with all necessary information to establish an effective SSC. This comprises at least the identification of required support equipment, software tools, and personnel with corresponding competence, technical documentation, facilities and infrastructure.

An SSC takes into account all activities to enable a continuous usage of software within a Product. To find a common understanding of software support, it is recommended that the customer and the contractor harmonize a standardized concept for software support documentation and include it in an LSA PP.

Analysis activities for software are like those for hardware, and they take into account software operational and maintenance requirements. Special operational aspects are of particular interest for Product users because of the influence on the day-to-day business. It is also necessary to take into account aspects covering software modification, such as bug-fixing and software improvement (eg, software upgrades).

1.2 Purpose

The means of LSA and SSA establish an appropriate support concept for such equipment. The results of the various analyses allow to ensure the customer's needs on supportability, readiness, and operability of hardware that contains software. The target readers of this chapter are IPS managers and/or LSA analysts who analyze Products containing software, whether they work for the contractors or the customers.

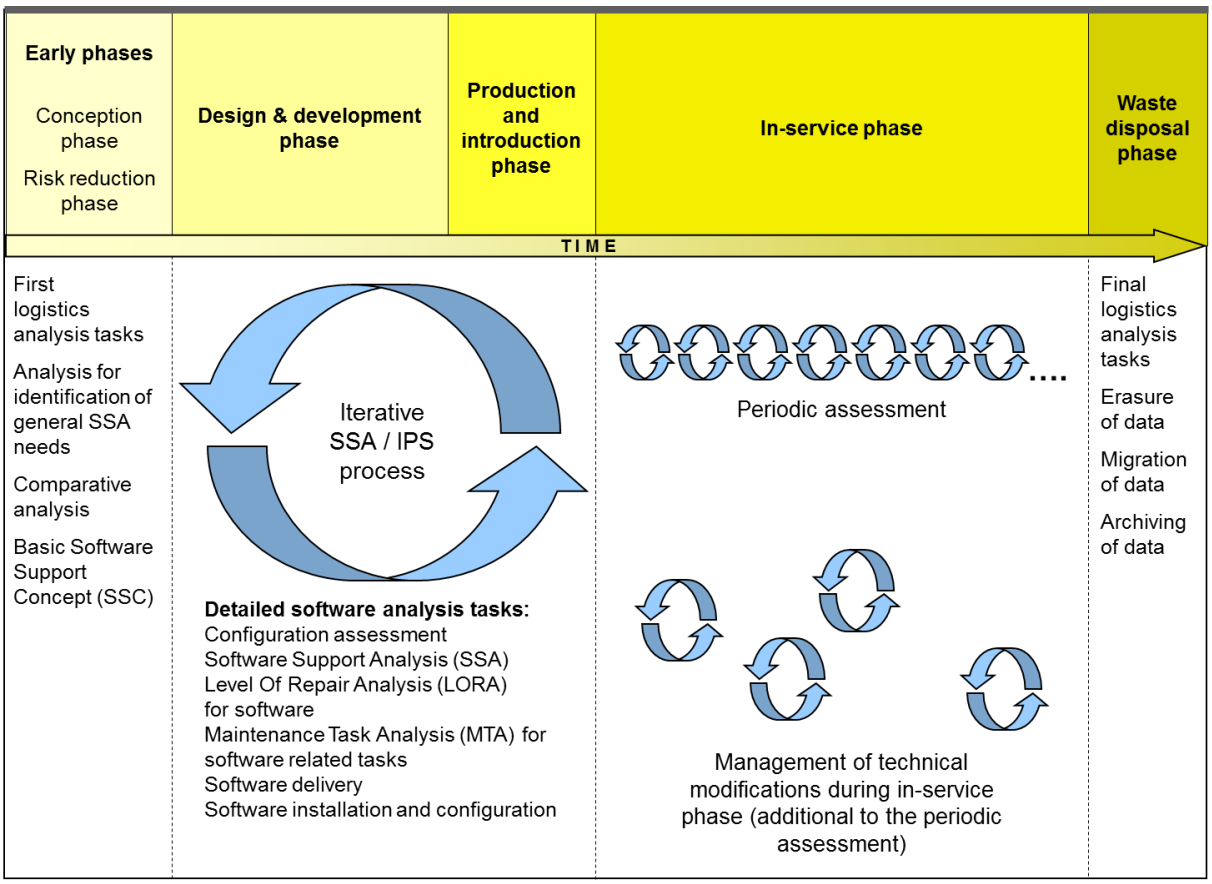
1.3 Scope

This chapter provides the supportability analyst with guidelines for handling the specific requirements concerning software with respect to maintenance and operation. This chapter clearly defines the interrelation between software and hardware, and explains the method to integrate software aspects into the overall LSA process.

With respect to software, it is necessary to make a distinction between operational/maintenance aspects and real software modification. For example, operational aspects can include the simple act of loading working software or required data to equipment within the Product, whereas software modification covers aspects that deal with the update of the source code to fix a problem, improve the performance of a software package, or adapt to changes in procedures, data, or systems that affect the software performance.

2 Software support analysis in the different project phases

Like the application of LSA, SSA is an analysis activity, which can be applied throughout the entire life cycle of a Product containing software. In the different life cycle phases various analysis activities must be performed. Refer to Fig 1. Depending on the phase, the importance of some activities can increase, while the importance of other activities decreases. For example, during the design and development phase, SSA is of high importance as it can influence software architecture and design in terms of supportability requirements for future needs to load, service, or modify the software packages. In an early concept phase, SSA helps to identify basic software support requirements. During the in-service phase, aspects like those for hardware are valid. Technical modifications will force the analyst to reassess the support solutions for the affected hardware/software. For this reason, the depth of an iterative SSA/IPS process will depend on the extent of the technical modifications. Basically, these activities are the same as those in the design and development phase, and in the production and introduction phase, but they are normally less extensive.



ICN-B6865-S3000L0062-002-01

Fig 1 Software support analysis in the different project phases

In the disposal phase, it is important to handle software and/or data during the disposal of hardware. It is necessary to ensure data handling complies with the project's requirements and it is mandatory to erase sensitive data from scrapped hardware.

For archiving purposes, it is necessary to preserve existing data from disposed hardware components. In case existing data is needed for further operation of a new Product, it is essential to analyze feasibility of data migration and establish a concept for data migration.

3 Breakdown concept

Chap 4 provides guidance on the development and assignment of a BEI for hardware and software with physical and functional breakdown examples. It is recommended to follow this guidance and the examples for the development of breakdown relationships between software and hardware.

It is possible to obtain a BEI for software like the BEI for hardware by following the concepts of physical or functional breakdowns. However, due to different characteristics, it can be difficult to associate software within the traditional physical/functional breakdown. This applies to complex systems (eg, modular avionics), where the physical location of a specific software can be unclear. Often, software is not a clearly defined physical entity, therefore it can be difficult to identify the appropriate indenture level for the software item within a physical or functional breakdown. Even in the case of a clear assignment of software to hardware, the question can arise as to where the software can be allocated:

- Is it part of the hardware element where the software is loaded?
- Is it part of the hardware element where the software is executed?
- Is it part of the hardware element where it physically is installed?

In this context, the breakdown is different from the traditional physical/functional hardware breakdown and it is important to establish a consistent approach. The physical breakdown can be used to identify software within a Product breakdown for operational purposes, and provide the necessary software view for traditional maintenance purposes. The functional breakdown is a software engineering view that shows the modifications and integrations to the software, and provides the relationships between the different software elements to be considered in case of redesign.

3.1 Functional and physical breakdown principles

Normally, operators can handle the software items identified within a physical breakdown. This software must be considered for operational aspects such as loading/unloading. The software can be documented at different indenture levels of the Product breakdown.

The allocation of physical software to a specific indenture level is driven by the need to maintain an adequate configuration control of the Product and to underline the supportability impact that said software can have on standard operation and maintenance. Traditionally, software was assigned to a hardware element. New configuration management issues and approaches have changed this concept and now, software can be represented by a (BE and/or a part/component within the LSA data. Refer to Para 3.3.

Software items identified within a functional breakdown are used to describe the functional/design aspects that are important for software designers, especially since this type of breakdown usually indicates the need for integration with other software, and it must be considered for software modification support. Therefore, it is important that the functional breakdown follows the software design as closely as possible.

3.2 Physical software categories

There are three categories of physical software:

- Field Loadable Software (FLS)
- Shop-Loadable Software (SLS)
- resident software (also called firmware)

The concept for the three categories is established in the configuration management principles that control the configuration of a specific Product.

3.2.1 Field loadable software

FLS is any software that can be installed on one or several pieces of equipment in a Product/system without the need to remove the equipment from its installation location. FLS is considered as an item within the Product breakdown. Therefore, FLS modification results in a change of the FLS part identifier and configuration of the system or Product where it is installed. FLS loading does not cause a change in the part identifier of the hardware. FLS loading affects the Product configuration because the function of the Product can change when using a different piece of software.

3.2.2 Shop-loadable software

SLS is any software that can be loaded into an LRU, but requires the removal of the LRU from its installation location on the Product. Replacing any hardware component is not required. A modification of the SLS changes not only the part identifier of the SLS, but can also change the part identifier of the LRU to which it is loaded. The same happens when a Shop Replaceable Unit (SRU) is replaced by a different one, to maintain the form, fit and function principle that drives configuration management (ie, the function changes when replacing software by different software). In this case, the change in the hardware part identifier affects the Product configuration.

3.2.3 Resident software/firmware

Resident software is any software that can be loaded into an LRU or SRU but requires the removal of LRU/SRU from its installation location on the Product and can also require the replacement of a component where the firmware is installed. Although resident software can have its own part identifier. When it is installed on a component or loaded onto an SRU, the part identifier for the component or SRU will change. This can also entail a change to the part identifier of the LRU at the next higher level.

3.3 LRU and SRU aspects

The recommended way to handle software within an LSA Product breakdown is to identify software as LRU, SRU or part/component, depending on the software category. This not only simplifies the overall Product configuration management, but it also clarifies how to handle the Product configuration when using FLS and there are no tags on the target hardware regarding the software it contains. This is not an issue when the configuration is handled at the next higher level.

Similarly, a software loading task due to a hardware repair can be integrated into the overall maintenance tasks because a specific software SRU or LRU is affected. In principle, it is not different from performing a task on a hardware SRU/LRU and subsequently performing additional tasks at the next higher breakdown level. Therefore, a software loading task after a hardware repair is similar to, for example, replacing a consumable.

3.4 Functional software categories

The functional software breakdown follows the software design requirements, as it is mainly oriented towards software modification. It is possible to use the classic Computer Software Configuration Item (CSCI), Computer Software Configuration (CSC) and unit level views. However, contrary to “traditional” software engineering, it is also necessary to provide the higher levels of abstraction (subsystem and system level), as they indicate interdependencies and need for integration, including the need for system or subsystem level integration rigs. In this context, software assembly items within a functional breakdown can support the grouping of functionality that is documented using the same principle as that for assemblies within physical breakdowns. It is possible to use said assemblies to provide a helpful additional level of abstraction without changing the overall structure and functionality of the software design.

It is recommended to use a BE revision identifier in the event of a release of a software version that significantly changes the software structure, functionality or supportability elements (eg, change of programming language of a specific module) that can entail a modification of the support tasks and/or support infrastructure. It does not provide any benefit to full analysis

performance because there is just a change in an algorithm, but it can be essential if the software class is likely to change.

3.5 Data

It is possible to handle data in a similar way to software, considering them from both the physical and functional point of view. The peculiarity of physical data is that they can be loaded/unloaded but also prepared, so it is not a pure software aspect. Software modification is a design activity, while data preparation can be considered an operational activity. However, the difference between the two activities is not always obvious.

In general, data can be included within the LSA data in the same way as software, including preparation, if applicable. They can also be included in the functional breakdown to ensure retention of configuration and dependencies. For example, a software modification can entail a modification to the mission preparation systems.

Together with the customer, it is necessary to discuss and decide whether to consider data as a software element, or as a special category within the LSA data. Though standards such as RTCA/DO-178B include data in the software definition. Treating data as a separate element for supportability purposes offers some benefits, eg, data is available as a BE to assign loading/unloading tasks.

4 Software support analysis

SSA is a consistent methodology to guarantee proper software supportability throughout the requirements, specification and design phases, to define the most cost-effective support concept that meets the operational and software modification requirements. It is necessary to establish an adequate support infrastructure before the Product enters the in-service phase. The main goals of an SSA are to establish supportability requirements concerning software in the early program phases, and to influence the software design to ensure supportability for both Product operation and later modification processes. The standard SAE JA1004 (Software Supportability Program) outlines a stand-alone software supportability program.

4.1 Software support analysis process - commonality with the LSA process

In general, the SSA process covers supportability aspects that are like those for the LSA process, as described in Chap 3. For Products that contain software as a basic element within one or many pieces of equipment, it is necessary to apply a number of software support activities, depending on the situations that trigger the need for support. In this area, there are two main categories of analysis:

- operational/maintenance aspects (eg, loading of data or software to the corresponding hardware, system recovery, data transport and archiving)
- software design aspects (eg, real modification of software for bug-fixing or improvement purposes)

When carrying out SSA, it is necessary to consider the relationship and differences between the two task categories, and the types of software items in the functional or physical Product breakdowns. Typically, support activities not involving any modification are analyzed against physical breakdown items. In all cases, LSA data include the analysis results concerning operational aspects, including the description of tasks such as the loading of data and/or software to the corresponding hardware element within the Product breakdown. After a change in the equipment, for example, it can be necessary to reload software and/or data to the new equipment or to configure it after or before installation into the Product. The use of hardware for special purposes can require special software or data packages to support this kind of operation. There are no changes to the software itself.

Normally, a software modification directly influences the software source code and involves modifications to the source code. Since the requirements for these activities differ from the

operational aspects concerning software, the relevant documentation is normally separated from the support/operational aspects. In case software packages are analyzed to determine their requirements for real software modification tasks, it is possible to treat said packages as SSA candidates similar to LSA candidates for hardware.

4.2 Software support analysis candidate selection

In general, SSA candidates are either physical or functional BE that are subject to any kind of SSA. Supportability significance drives the candidate selection. Due to the two main categories of software support activities, candidate selection occurs separately for each support category.

4.2.1 Selection of physical candidates

4.2.1.1 Software

SSA candidate selection must take into consideration all software items that the operator can load and/or install separately. The hardware item carrying the software can be a candidate item, too.

For distributed software, the candidate can be a subsystem or a system within the Product breakdown. Typically, loading and/or installation tasks are documented against the corresponding hardware. It is possible to document other operational tasks against a software BE as well (eg, transportation or distribution of software).

4.2.1.2 Data

Data are usually electronic in nature, and handled similar to the corresponding software. Therefore, it is possible to analyze together the support of software and dependent data (although not in the same way). It is necessary to analyze each SSA candidate for special support aspects for data. The analysis includes, but is not limited to:

- Does the required data preparation process need special software or hardware?
- Is a data validation process required?
- Are special transmission media or networks required?
- Are there special security aspects to data loading?
- Are there special compatibility requirements with the existing executable software?
- Are there special requirements concerning size and format (eg, databases, file formats)?

4.2.2 Selection of functional candidates

An SSA candidate identified in a functional breakdown must take into consideration all software items that are subject to software modification activities. It is possible to use the following questions to analyze each software item to decide whether it is subject to a software modification activity:

- Are any support initiators expected for the software items that require a modification of the software source code?
- Does the software item in the breakdown include all software within a functional system or only within a physical equipment?
- Does the software item use a separate design?
- Does the development of the software item require special hardware and/or software tools?
- Does the software item contain proprietary software, such as run-time libraries or COTS elements?
- Are there different versions of the same software item for usage on different platforms?
- Are there deviations of the software items to the general design environment?

4.3 Task requirements for software

Like the LSA methodology for hardware, the identification of any task requirement that initiates maintenance activities is the starting point to document the relevant maintenance aspects. It is possible to describe these events as "software support initiators" and to apply the event-driven methodology to SSA. However, the events concerning software are somewhat different from the events for hardware. They can be grouped according to the implications of:

- operational events
- technical events
- software improvement requirements
- software failures

4.3.1 Operational events

This type of event relates to operational requirements. Operational events can be documented within the related hardware item. The same applies to the corresponding operational task. Typically, the covered aspects are:

- Corrective or preventive maintenance tasks relevant for the computing hardware
  After hardware maintenance, different types of tasks can be necessary (eg, reloading of data and/or software, installation and configuration of software packages on new installed equipment).
- Use (mission) preparation
  Besides the installation of additional hardware, the preparation for use of a Product can require the installation of supporting software packages or loading of special mission data.
- Post mission requirements
  After Product use, it is necessary to unload or archive the data created during the mission. Moreover, it is necessary to uninstall software packages required for a special mission, and reconfigure the Product to a standard configuration.

4.3.2 Technical events

Besides hardware maintenance, other technical events can trigger software support activities. These events normally require adaption of the existing software packages to the modified environmental preconditions. Typically, the covered aspects are:

- changes to the parent hardware
- changes to the parent software (eg, upgrade of operating systems, corresponding database systems or firmware)
- changes in the technology of interoperating systems
- changes in the technology of network interconnectivity

4.3.3 Software improvement requirements

Typically, software undergoes a constant improvement process. Normally, software release changes take place at regular intervals, depending on the complexity and size of the software package. Besides fixes to real software failures, the primary inputs for new releases are dedicated user requirements or changes in the environmental preconditions. In this context, improvement means the introduction of new features that improve or extend functionality, or the usage comfort of an existing software package.

4.3.4 Software failures

Failures initiated by software are main support initiators. The reaction to these failures depends on their severity. Not every failure caused by software will initiate an immediate software modification activity.

For software failures, it is recommended that a classification concerning severity be introduced. Refer to Table 2. For each failure class, it is possible to define a sequence of activities within an SSC. In every case, proper documentation is required to support later diagnosis.

Table 2 Failure classes

Class	Description
Minor	In general, these failures concern user comfort. In many cases, these events are not real failures, but rather a deficiency in the design of software features. In some cases, these minor failures can be actual failures, but other software features, or an alternative handling can solve them. However, it is recommended to collect, document and report these minor events to the software support organization. This information can support the incorporation of improvements into a future release of the software package. Especially the correction of marginal insufficiency can considerably increase user comfort and acceptance.
Medium	These events cause decreased Product functionality. If this type of event occurs, the user is not able to continue using the Product as expected. However, the Product as a whole works without severe disturbance (eg, no need for shutdown). It is possible to continue using the Product with reduced performance or even with no restrictions at all. These events normally need a timely correction, or at least a defined workaround to deal with the failure until it is finally corrected. Normally, a software modification is necessary.
Major	A major software failure is an unacceptable event. This failure causes the shutdown of the entire Product, or the necessity to operate the Product under restricted conditions. The normal capability of the Product is significantly degraded. These events need correction as soon as possible. In case of emergencies, the software support organization must react as soon as possible to restore the entire Product to full operation. It is necessary to discuss and determine the reaction times together with the customer.

It is not possible to treat software failure exactly like hardware failure. In the case of a hardware failure, it is possible to clearly define any relevant maintenance activity, such as a repair by replacement of components or a replacement of the LRU itself. In case of a software failure, it is normally not possible to identify the required activity immediately. For example, a restart of the Product can be sufficient, and the failure will not occur again. However, other types of failure can be severe and lead to Product shutdown. In this case, it is better not to restart the Product, to avoid the repetition of the failure and a further corruption of the entire Product.

Corruption of data or the executable code can be another important aspect. A virus infection or corrupted carrying hardware (eg, the loss of functionality of a hard disk) can lead to the corruption of data and code. In this case, there is no real inherent failure in the software source code. This event is handled as an external damage.

4.4 FMEA/FMECA aspects

Since FMEA or FMECA, respectively represents an analysis method for the complete Product (covering hardware and software aspects), the results of a technical FMEA/FMECA are relevant for software. For any identified failure mode, it is necessary to determine whether the associated functionality depends on or is provided completely by software. During the design/development of the software, it is possible to use the information on this failure mode to design a software architecture that eliminates, or at least mitigates, the possibility of software failure that can cause a specific functional failure of the Product. In addition, the design can be arranged to ensure that any such potential software failure is detected and recorded, including its associated information, and that the Product continues operation in a safe mode.

During the in-service phase, Product downtimes due to software failures drive the software modification activities. A principle of the failure mode grouping (refer to Chap 7) is to group together all software failures that lead to a specific system failure and to group together all specific system failures that lead to the same request for software modification. The analysis depends on the function that the software performs.

During the in-service phase, it is necessary to provide information to the software engineers on specific failures, and to avoid unnecessary maintenance tasks due to software failures. The possibility to identify and localize a software-specific failure for example by built-in-test capability can avoid an unnecessary hardware removal/replacement (eg, a software failure cannot be rectified by replacing the equipment) and cases of "no failure found" related to hardware.

4.5 Preventive maintenance analysis for software

PMA methods such as S4000P or Reliability Centered Maintenance (RCM) are not applicable to software packages in the same way as those for hardware. Software failures are the result of unintended effects of the software design. PMA is not effective in avoiding such failures, since they result from design shortcomings. However, some software failures can affect safety. The results of PMA carried out for a piece of hardware containing software can identify preventive maintenance tasks that are operational in nature and concern software (eg, a mandatory software test before a mission). However, it is still a preventive maintenance analysis for hardware taking into account software elements as potential failure causes.

A PMA on the software itself can result in a periodic assessment of the software during its life and not in preventive maintenance tasks performed on the relevant hardware. Software shows a bathtub curve for failure, with many failures, initially. Bugs are fixed with successive software releases leading to a stable behavior. During later releases, failure rates can increase again due to the increasing degradation of the software, modified beyond its initial scope. A periodic assessment of the failure rate can determine whether the software is reaching its end of life:

- due to increasing software failures because of too many modifications and alterations
- due to failure of the underlying hardware or operating system to process the added functionality. This can lead to either a software rewrite or perhaps a complete equipment or even system redesign.

4.6 LORA aspects

A LORA procedure (refer to Chap 11) can identify a proper maintenance level for a software related support task. With respect to operation, the performance of support tasks related to software or data loading can be covered within a LORA for the hardware-related tasks, or at least by a similar process.

For software modification, detailed information about equipment (software development environment concerning hardware and software developer tools) and adequate personnel competence is required. In addition, contractual warranty and software ownership can influence the identification of an appropriate support level for software modification activities. For management tasks, the process also becomes important to determine the best location to perform tasks such as the production of software or data media.

4.7 Software support tasks

After the identification of the relevant task requirements (software support initiators), it is necessary to identify the appropriate tasks (operational support/maintenance tasks or modification tasks). The goal is to define all operational, maintenance or modification tasks, their related resources, and additional task characteristics such as duration, manpower requirements, preconditions or safety conditions.

SAE JA1005, Software Supportability Implementation Guide, provides guidelines to the analyst regarding which type of software support tasks can occur depending on different task requirements. Refer to Para 5.

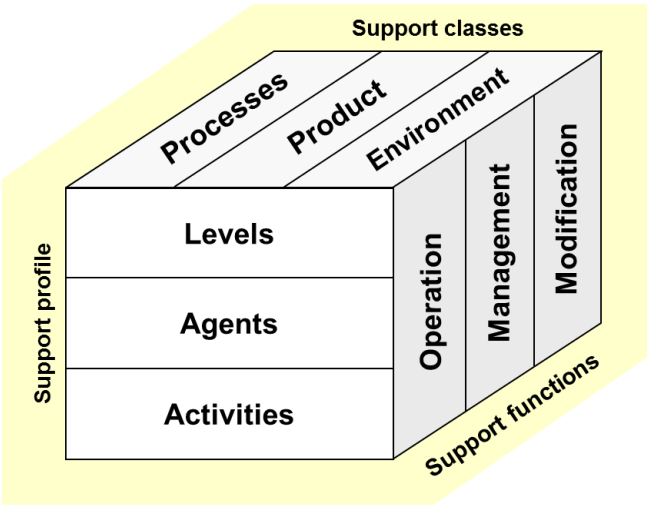
5 Software support concept framework

Achieving a proper supportability framework is one of the most important design principles for software, in a similar way as for hardware. To this end, it is required to establish an SSC at an early stage of the software design process. The earlier the design pays attention to support requirements, the better the supportability will be.

It is recommended to give a guideline to the supportability analyst using this specification. The guideline must cover all the important areas concerning software support and must consider contractual agreements between the customer and the contractor. All aspects can be tailored to the required extent of the project.

Note

An SSC can have different perspectives, all interconnected and influenced by each other. Refer to Fig 2.



ICN-B6865-S3000L0063-002-01

Fig 2 Aspects of an SSC framework

The extent of a software development or procurement project determines whether the process to establish an SSC can be included in the overall LSA process (refer to Chap 3). The procedures of the general LSA process can cover the aspects of software support. Relevant documents such as an ORD or CRD can also cover software support aspects concerning the required usage information. The LSA GC also covers the software support aspects and the relevant contractual agreements, documented in the LSA PP and/or LSA GD. The candidates for a detailed SSA can be added to the CIL. It is also possible to include the rules for SSA candidate selection in the LSA GD.

For complex Products with a considerable amount of software, it is recommended that software supportability be treated as a separate topic. It is possible to create and implement a separate SSA program plan, and to hold a separate GC for software aspects. However, it is necessary to consider the interaction between hardware and software development. Software is designed to address functionality of hardware components. Therefore, both must work together, and it is necessary to harmonize, document, review and put into practice processes that guarantee proper cooperation. Refer to SAE JA1006, Software Support Concept.

5.1 Software support profile

The software support profile (refer to Fig 2) takes into consideration the location of the software support, the organizations and individuals that are the relevant stakeholders in a software

support environment. Additionally, it is necessary to discuss and agree with the customer the processes regarding how the different instances work together in an integrated way.

5.1.1 Support levels

For software, the levels of support can be addressed in a way similar to the maintenance levels for hardware within the LSA. The support levels for software can differ from the hardware support levels and can be used as input to the LORA decision process for software. In addition, classification is similar to the maintenance levels for hardware, but some special aspects concerning software can be considered additionally. Table 3 provides examples for software support level definitions.

Table 3 Examples for software support level definitions

Support level	Description
Level 1 (operational level)	This support level describes the location at which software is in operational use. Within this level, normally only simple support tasks can be carried out. These tasks are typically covered by the operational support.
Level 2 (intermediate support level)	An intermediate support level can cover all support activities which are related to operational support but also to management support. User support in the form of a help desk or a hotline can also be covered by an intermediate support level. Activities can be software upload/download, firmware update or installation of patches. Typically, the intermediate support level for software is located near to the operational area but is not part of the operational site.
Level 3 (high support level)	This support level is normally located at a central site. It can cover activities from the management support and modification support. Software packages characteristics (eg, complexity, size, modularity, programming techniques) and available equipment (eg, tools, personnel, facilities and infrastructure) of support sites determine the extent of the capability to modify software packages.
Level 4 (software vendor and/or original developer)	Vendor level is normally the highest qualified support level for demanding tasks concerning software modification. In many cases, the vendor is the original developer of the software, and has competent personnel and all necessary developer tools available. This level is often selected for modification support because the user itself does not have the required capabilities for these complex support tasks. Additionally, it is necessary to consider aspects of responsibility and liability, which can force the user to address software modification to the original vendor.

5.1.2 Support roles

As a first step to define software support roles, it is necessary to clarify customer (buying department and users) and supplier (vendor and technical support personnel) roles. As a second step, it is required to allocate the roles concerning software usage and support to personnel who can offer the required services. These roles can be:

- simple users without any rights to modify or recover any software system (not a real support role, but in general only a service receiver). The main task of a simple user is to keep the software operationally running and report any software-related failure.

- power users with some basic knowledge and the ability to perform a system recovery. With respect to software support, this role can perform simple operational support actions to keep the software running, such as loading mission data. Normally, a power user does not perform any software modification activity.
- system administrators, who are power users on a higher level. They must be able to cover all activities concerning operational support and eventually management support. System administrators handle direct support for simple users or power users who work with the software packages. Normally, a system administrator is not involved in any software modification activity. Typical system administrator activities include software configuration, for example changing performance parameters, granting of user access rights, loading additional software packages, or installing updates.
- service providers, who can provide qualified operational support and additional services, such as a hotline. Service providers can be located onsite with the customer and vendor, and in some cases, are able to provide modification support.
- vendors or original software developers, who can provide modification support up to the highest support level. Vendor support is normally located off site.

All these roles must be equipped with the required competence and access rights necessary to carry out their tasks. The SSC describes all roles, including their profile. It is necessary to define and document the limits of competence clearly, and all affected personnel must accept them.

5.1.3 Support scenario

For an appropriate planning of a software support scenario, it is advisable to describe use cases for the different support roles and support levels, indicating how the different resources at different locations work together, and which processes are the basis for the cooperation. The customer and contractor must agree, implement, and control this support scenario.

5.2 Software support classes

The quality aspect of an SSC includes all objectives involved. The software support classes ensure compliance with the rules by means of implemented and used quality standards.

5.2.1 Processes

The customer/user and the software support provider must discuss all established processes and provide written documentation on the implementation of these processes. The documentation must clearly address process characteristics including inputs, outputs, controls, and resources:

- Is there any written documentation of the process (eg, flowcharts, descriptions, requirements, responsibilities)?
- Is there an accepted standard to describe the relevant processes (eg, international or company internal)?
- Are there any measurable parameters for performance control, and how is performance control measured?
- Are inputs and outputs, procedures, requirements and effective control mechanisms defined and documented?

5.2.2 Product

It is advisable to address supportability aspects as early as possible in the software Product characteristics. Inherent quality characteristics of software depend on the level of quality of software design. Requirements for a good software design include many aspects, starting with the software specification and ending with a perfect user handling, supported by a proper graphical user interface. Typical quality aspects are:

- modularity
- changeability and expandability
- simplicity and easy handling
- stability and consistency

- performance concerning processing speed
- instrumentation (in the sense that the software is purposely instrumented for an understanding of how it is operating)

If the software development process considers these aspects in detail, it will be easier to handle and support the software package during operation. However, even well-designed software can cause problems if the implemented support concept is insufficient.

5.2.3 Environment

Environmental aspects affecting quality start with the personnel involved in software use and/or support. To implement proper software support, it is essential to address personnel characteristics thoroughly:

- Who is the right person for a software support job?
- What are the required skill levels and experience?
- How many persons are required to guarantee a trouble-free software support?
- What is the level of motivation of the personnel?
- What employee turnover can be expected?

Not only personnel aspects influence the quality of the environment. Simple aspects such as the quality of facilities and infrastructure, computer workplace equipment and size of offices can affect the quality of the support services.

5.3 Software support tasks

Support tasks related to software fall into categories depending on:

- Is the task a simple software loading/unloading task, or a simple software transfer task (eg, from a device A to a device B)?
- Is it necessary to remove hardware from a Product to load/unload software?
- Is the maintenance task associated with any case of failure recovery or documentation of software problems?
- Is any case of software modification involved?

Note

Software modification leads to a wide area of specific activities, which software design and development activities must address in detail.

5.3.1 Operational support tasks

The operational support tasks describe all activities associated with the day-by-day operation of the Product. However, the quality and depth of impact of these tasks on the usage of the Product (eg, downtime, test requirements) can vary. In general, these are rather simple tasks carried out by a simple user or a power user on a low support level, which only require the software package to be available on a compatible medium, plus a corresponding user manual. It is necessary to take into account different methods of installation. For stand-alone Products, system administrators will have different requirements compared to those of large computer network systems. In modern network architectures, software and data installation and loading is carried out with the help of software deployment tools.

Table 4, Table 5 and Table 6 give an overview of the types of tasks that can occur, including further definition of some terms concerning the handling of software packages. These tasks fall under three subcategories:

- tasks for the provision of new functionality and/or ability
- tasks for Product recovery and for problem reporting in case of software failure
- organizational tasks

Table 4 Tasks for the provision of new functionality and/or capability

Type of task	Description
Software installation	Software installation indicates an installation routine for a software package, leading to the provision of a possible functionality by means of software.
Software de-installation	De-installation of a software package indicates a de-installation routine or deletion of files from a storage device, leading to the definite removal of a possible functionality realized by the mean of software by carrying out a de-installation routine or by simple deleting of files from a storage device.
Software loading	Software loading is a task that goes beyond installation and therefore, it is more complex in many cases. In addition to the installation routine, which is part of the software loading, it is necessary to consider software configuration.
Software configuration	In this context, the term configuration means the change of internal parameters (eg, operational or performance parameters, editing of configuration files, granting of user access rights, path information, database connections) for software operation. These activities do not require any new installation of the software package. The functionality of the software package is sufficient to change the configuration parameters of the software. For the configuration, it is important to have sufficient access rights (eg, administrator account to log into a software) to introduce the changes.
Data loading	To provide special capabilities to the Product, data will be normally loaded using the existing functionality of an installed software package. In this case, there is a fine line between software and data, depending on the type and format of the data to be loaded. Typically, data loading tasks are used to prepare operations or missions. Data do not change the basic functionality of the installed software package. For example, the maps in a navigation system can be considered as data. The software of the navigation system uses these data, for example, for the graphical representation of a city map. When loading another data set (eg, package of another country), the functionality of the navigation system will not change.

A particular aspect of installation and loading of any software/data is the embedded software or firmware respectively. In this case, the allocation of new functionality or capabilities can be more demanding. Transferring new software code or new data to the storage devices of the embedded software (eg, resident storage chips on a computer main board) normally requires some special tools and support equipment (eg, to gain access). The MTA of the installation or loading tasks for embedded software or firmware identifies the relevant requirements.

After each change of software by means of an installation/loading procedure, it is recommended to define the methods that will ensure proper Product functioning. It is necessary to investigate thoroughly and, if required, document (eg, in the installation instructions) the need to have adequate competent specialists available to perform a functional test and confirm proper work.

Table 5 Tasks for Product recovery and for problem reporting

Type of task	Description

Type of task	Description
Software recovery	Software recovery includes all basic diagnostic activities and simple recovery actions, such as system reboot/restart after a system or software shutdown. Normally, low-skilled personnel can carry out these activities safely. Even in case of a successful recovery to full operational capability, it is strongly recommended to perform a diagnosis of all potentially relevant components of the Product. Documenting thoroughly all available data concerning the problem occurrence and the recovery actions can be useful for evaluation purposes. This can lead to several outcomes to rectify the cause of the problem immediately, or to ensure full operational capability by alternative means. In case it is not possible to solve the underlying problem within a short time, a temporary downgrading of operational capability can be considered.
Recovery problem reporting	In the event of a failure involving software, it is recommended to generate a problem report, even after a successful simple recovery (eg, restart or reload). This problem report contains at least: – Product condition/configuration/version (software and hardware platform) at the time of failure. This information is important with respect to fault investigation on transitory effects, as it can be necessary to be able to reproduce the failure effect in an appropriate test environment or reference system. – type of software failure – outcome of software recovery task (successful, negative, partial) – classification of failure severity It is necessary to transmit problem reports to the responsible personnel who can analyze the problem to establish its priority, and determine any workaround or need for temporary operational limitations.
Post operational data extraction	Software support tasks carried out immediately after operation are often associated with data extraction (eg, for operational and/or engineering analysis) or fault investigation. With respect to data-related functions, a range of data administration tasks can be required (eg, provision and upkeep of data as routinely needed by a software program to fulfill its operational role). Fault investigation tasks will relate to activities on the primary system and to the deeper-level testing on remote reference systems, which replicate or simulate the primary system’s environment.

All tasks related to recovery of a Product after a software failure do not normally contain any software modification activities. However, recovery of a Product can require the involvement of an updated software package.

Problem reporting can be a starting point to identify the need for modification. Even if recovery tasks were successful, it can be necessary to modify software to rectify the failures that occurred, thus avoiding future repetition of the failure.

Table 6 Organizational tasks

Type of task	Description
Software delivery	Software delivery indicates new software received by the user from software suppliers. Receipt of new software implies several activities including quality and validation aspects: – checking condition and completeness of delivery – checking configuration status information – checking license information – checking compatibility between delivered software and impacted Product components
Operational configuration control	This task takes into consideration the compatibility of software with special hardware. Sometimes, a specific software package only works on certain variants of a hardware item. On the other hand, it can be possible to install/load different software packages to equipment, depending on the required functionality. The user must know what is compatible. It is mandatory to avoid configurations that are not tested or even not allowed.

5.3.2 Management support tasks

The management support tasks cannot be considered as pure operational support tasks. Software modification is not part of these tasks either. This kind of support is located somewhere between the operational support and the real software modification tasks. Users and technical support organizations can perform these activities across all levels of support. Refer to Table 7.

Table 7 Management support tasks

Type of task	Description
Problem reporting	This type of problem reporting covers more aspects than the recovery problem report. In this area, it is essential to implement and upkeep a management system for failure reporting and corrective actions (eg, a bug tracker system). Within this system, all actions are monitored and organized: – prioritization related to the severity of problems – request for software modification – monitoring of software modification and bug fixing procedures Potential additional aspects are the continuous monitoring of system effectiveness and the development of support costs. To ensure best software performance and supportability, it can be necessary to change software because of these kinds of problems.

Type of task	Description
System configuration control	In addition to the operational configuration control, the system configuration control is becoming increasingly important. The interconnection of many computer systems in large networks within a complex Product requires configuration control at a higher level. The first step is to ensure that software packages are only installed on compatible hardware equipment. Operational configuration control covers this step. The second step is to ensure that all Product components carrying software/data and connected to each other work together properly. System configuration control covers this step. In modern Products, this can become a complex challenge. It is strongly recommended to establishing a compatibility matrix for software/hardware. Only tested configurations of an assembly of hardware and software components can be operated under safe conditions.
Delivery and installation	In the context of management support tasks, the contractor or the supporting organization carries out these activities. In particular, after the modification of software and the creation of a new software release, the modifying organization will focus on the best way of providing the new software Product to the affected users, whether they are a single client or thousands of clients in case of widespread Products. Packaging and distribution paths are of special interest in the case of sensitive or security-relevant software.
User support	User support covers a wide range of activities carried out by the supporting organization. The interactions between the technical support organization and the user can be: – support for installation, set-up or operation of software – user help desk/hotline – user queries, which are answered by the supporting organization by providing result reports – user training

5.3.3 Software modification tasks

These tasks refer to a longer-term process that implements changes to software. Normally, a supporting organization of the contractor at a high level of support (manufacturer or software vendor) carries out these activities. However, also a customer can modify software, particularly lower-criticality changes. This requires corresponding equipment, as well as the appropriate competent personnel. Software modification tasks cover the software modification activity itself, which includes coding, change implementation, and testing, as well as the related tasks, such as problem investigation and configuration control. Refer to Table 8.

Depending on the reasons suggesting the need for software change, said changes can be:

– corrective changes
– adaptive changes
– perfective changes

In general, the most critical reason for software change is the occurrence of real failures, which leads to an unacceptable situation within the operation of the entire Product. Such a failure will normally lead to a corrective modification of software. Adaptive changes can be necessary due to a critical situation. However, predicting and planning this kind of changes is usually easier than performing corrective activities due to a real software failure. The time schedule for

adaptive changes is not normally as critical as that for a corrective task. However, in modern Products, it becomes more and more difficult to predict the adaption needs for software over a longer period.

Perfective changes aim to increase the user-friendly properties of the software or software functionality. Normally, the implementation of these changes occurs through an upgrade concept. The requirements of the users will be collected during a certain period and implemented within a new release of the software package.

Table 8 Software modification tasks

Type of task	Description
Problem investigation	The initial concern of problem investigation is usually to determine how to further operate the affected software taking into account the actual failure (eg, with lower capability). It is necessary to detect the presence of a real software bug or a real technical problem. For example, triggering any real software modification because of user mishandling must not occur. After identifying the problem correctly and putting in place any appropriate interim solution, an analysis will follow on possible changes to software to solve the problem permanently. For a proper problem investigation, adequate resources and tools for failure finding and documentation must be available. This can be a simple programming environment with certain compilers and debuggers, and/or a complex environment that allows simulation and failure reproduction.
Change implementation	The implementation of changes into software packages must consider some crucial aspects. The supporting organization usually hosts a complete repository of the entire software development process, from the beginning to the actual status. Coding, compilation and testing of software requires special tools, which must be available. Additional personnel with appropriate competence is necessary. It is necessary to take into account that any basic change in technology can cause support problems. Examples can be obsolescence or the superseding of tools that are required for software modification, or the change of basic operating systems on computers. This can have a large impact on costs or even require the upgrade of entire software packages in case the former software package is not compatible with the new environment. In the SSC, it is required to define and document the relevant standards and procedures that are used.
Change release	A new release of a software package marks the end of its change implementation process. At this point, after the installable and executable files/code have been tested and certified, it is possible to deliver them to the customer/user. All additional information is also available for the user, such as installation instructions and modified documentation. The supporting organization must ensure that the distribution of modified software releases is well organized. This is part of the overall SSC.

Type of task	Description
Configuration control	Configuration control of software during modification includes managing individual change development and controlling different software versions and documentation. For this reason, it is strongly recommended to use a software development repository to ensure compatibility between the new software packages and the existing hardware/software configuration. In the SSC, it is mandatory to define and document the relevant standards and procedures.

5.4 Software support concept framework - summary

In the frame of maintenance (corrective and preventive) and operational support for hardware, there is a clear understanding that all identified support tasks in this context do not modify the Product design. Typical maintenance tasks like repair, replace, calibrate, clean, or lubricate can be performed by the operator of a Product and which do not include any change in Product design. The same holds, for example, for inspection, test, fault location, remove/install, disassemble/assemble, transportation or storage tasks.

Any change in hardware design must be considered in the context of Product design and development, and in the context of Product support.

The situation is similar for software. However, the activities described in Para 5.1 to Para 5.3 fall under both software support activities, and software development activities. Those tasks are primarily relevant for supportability analyses for software, and usually they are maintenance or operational support tasks performed by the customer/operator. Those tasks are:

- In the context of operational support tasks, refer to Para 5.3.1
  - tasks for the provision of new functionality and/or ability
  - tasks for recovery of a Product and for problem reporting in case of failures
  - organizational tasks
- In the context of management support tasks, refer to Para 5.3.2
  If customers/operators carry out the management support tasks, they must be considered in the support concept for software. Therefore, it is necessary to include those tasks in the analysis process for maintenance (corrective and preventive) and operational support (eg, identification of corresponding task requirements and MTA), and finally include them into a technical publication.
- In the context of software modification tasks, refer to Para 5.3.3
  In this context, modification always relates to changes in the source code. Usually, customers/operators do not perform these software modification tasks. Therefore, those tasks do not fall under the scope of the analysis process for maintenance (corrective and preventive) and operational support. For example**,** a software bug fixing activity is not a "repair" of software in the context of maintenance**.** It is a software design activity because the source code will undergo modification.

  Note

  In some cases, and under project-specific circumstances, software modification activities can also involve customer personnel. In these cases, the customer must ensure the availability of all required resources and know-how (eg, competent personnel, IT-infrastructure, databases, network connectivity, software development environment, test tools, well known software development processes).

6 Supportability factors

To establish proper software supportability, the analyst must consider a range of factors that can influence software supportability, and consider all of them when determining the project requirements. Some of the factors relate more to operational support, while some others relate to software modification support. Generally, these supportability factors are associated with both software and its associated development process, or to the environment for software operation and support. Special projects can have additional aspects only partially covered by these factors. However, this is a good starting point for any analysis concerning supportability aspects of software. The supportability factors are:

- Compatibility matrix (refer to Para 6.1)
- Deployment (refer to Para 6.2)
- Documentation (refer to Para 6.3)
- Loading/unloading and installation (refer to Para 6.4)
- Transportation on hardware carriers (refer to Para 6.5)
- Expansion capability (refer to Para 6.6)
- Modularity (refer to Para 6.7)
- Modification frequency (refer to Para 6.8)
- Recovery (refer to Para 6.9)
- Safety integrity (refer to Para 6.10)
- Security (refer to Para 6.11)
- Size (refer to Para 6.12)
- Competence (refer to Para 6.13)
- Software distribution (refer to Para 6.14)
- Standardization (refer to Para 6.15)
- Technology (refer to Para 6.16)

6.1 Compatibility matrix

In systems where multiple equipment elements containing software work together in a network, it is important that equipment compatibility is guaranteed. It is necessary to ensure that different versions of software on different hardware items can interoperate properly in a networked system. On the other hand, the compatibility of different versions of a software package with different hardware items is also of interest. It is recommended that these compatibilities be included in the following documents:

- Software to hardware compatibility can be documented within the LSA data within the Product breakdown containing corresponding software parts and hardware parts. The relationship between hardware parts and software parts can be established by the usage of parts lists (Bill of Material, BoM). The possibility to install different issues of the same software package can be covered by the substitute part approach.

- A compatibility matrix concerning networked equipment carrying software packages, which must document the allowed configurations of the complete Product concerning compatibility of software versions/issues is required. It is recommended to establish this matrix outside of any LSA/SSA data.

6.2 Deployment

Software deployment is a crucial aspect for software supportability. It is necessary to provide sufficient support to each location where software packages are in use. The customer and the support organization must address in detail and agree on the methods to link the locations to support capabilities (eg, installation support, user helpdesk, troubleshooting, and recovery support).

6.3 Documentation

In general, documentation refers to two main aspects:

- supporting documentation for the software developer
- supporting documentation for the software user

  - end user documentation (user handbook)
  - administrator documentation (installation/update and configuration handbook for administrative purposes)

To ensure software supportability for the developers, documentation must conform to an agreed standard and be available to the organization responsible for delivering software support. It is necessary to document any software tool (eg, software development repository, compiler, bug tracker, test tools) used for software development and arrange their life cycle support.

For operational aspects, the documentation for an administrator or for a power user is of particular interest. The documentation must contain installation and configuration manuals, as well as troubleshooting instructions in case of malfunctions that can be rectified easily. In case of more severe malfunctions, the documentation must contain the relevant data for the supporting organization that must be contacted.

6.4 Loading/unloading and installation

The duration of software loading or installation is an important aspect. Therefore, it is necessary to optimize the software and corresponding loading devices for a proper and fast-loading process to take place. The same applies for the unloading of data or software after special missions or Product use. If it is necessary to download a large amount of data from a Product after a mission, a fast and easy download procedure must be available.

Safety is an additional aspect of software/data loading. Appropriate test procedures must ensure that loading or installation was successful, and that the Product works properly.

6.5 Transportation on hardware carriers

There are aspects that must be addressed if it is necessary to transport software using hardware devices. The type of carrier that will be selected depends on the type, size and safety requirements of the software. Aspects can be, but not limited to:

- Is it possible to transport the software on chip cards or USB sticks?
- Do you need a DVD or even a hard disk to carry the software?
- What are the safety aspects (eg, is it allowed to transport the software on a carrier, where it is easy to modify the files)?

6.6 Expansion capability

Expansion capability is an aspect of system design. Though related to the software and sometimes influenced by the software design, it is usually a hardware characteristic to be considered within the overall system design. It deals with the extent to which it is possible to modify software without being limited by restrictions on computing resources.

Examples of such constraints on computing resources are:

- available memory
- processor performance
- mass storage capacity
- input/output bandwidth
- database capacity
- network performance

It is necessary to consider expansion capability in detail, otherwise software modification will be limited, modification costs will significantly increase and/or additional activities will be required. Expansion capability is important in systems with fixed hardware configurations or embedded real-time applications.

6.7 Modular software design

Modularity is an attribute of the structure of a software design. A modular approach can help structuring different areas of functionality within a software package. The choice of design method, programming tools and programming language determine the possibilities to apply modularity to a software design. However, in general, the optimum approach to achieve modularity will balance functional and performance requirements against the need to provide a comprehensible and serviceable design.

Insufficient modularity can result in increased modification effort and costs because of the need to implement consequential modifications in different parts of the software.

6.8 Modification frequency

Modification frequency (also called change traffic) measures the rate at which software modification is necessary. Before the software is in use, estimates of the modification frequency are possible by comparing it with similar applications (eg, any data available from comparable in-service Products on change traffic and effort will be of significant value). It is possible to use some calculation methods based on the characteristic properties of the software to estimate the change traffic. Investigations can also include the experience earned during test and trial phases. Modification frequency influences stability, software integrity, and Product availability. Change traffic significantly affects the volume of software support activity.

High modification frequency involves a higher risk of downtimes of the entire Product because of implementation problems or even because of bugs from new software releases. It is essential to find a balance between the frequency of new software releases and the needs from adaptive, perfective, or corrective requirements.

6.9 Recovery

In case of any Product downtime caused by a software failure, it must be possible to reset the Product to full functionality or at least to an acceptable level of degraded functionality in an acceptable amount of time. It is necessary to consider this aspect in the early phases of software development to provide a proper instrumentation for recovery purposes. This can include, but not limited to:

- state of health monitoring
- fault-tolerant handlers
- design for debug features

This aspect is particularly important in systems deployed to space or other exposed locations. If software fails in such distant systems, it is crucial to have a potential recovery capability using the available instrumentation for real-time operational upgrade (eg, download updated software and/or data).

It is necessary to document recovery procedures completely, and the responsible users and administrators must know them well. Testing of recovery procedures must take place before a real emergency occurs. In this case, a clear understanding of what to do and how (eg, critical data can be restored from a data backup) must be in place. The best backup is worth nothing if it is not possible to retrieve stored data during an emergency because of unforeseen circumstances. It is recommended that personnel be trained to react properly, and that proper function of recovery be regularly simulated.

6.10 Safety integrity

The safety criticality of the functions that a software package provides determine the required safety integrity for the software package. It is recommended that the overall safety criticality of a system be analyzed by means of an appropriate hazard analysis technique. The partitioning of system functions in the system design derives from the criticality of software items. Design must ensure to minimize software items that implement critical functions, or at least segregate them from other parts of the software. System requirements must define safety criticality categories

and the levels of software safety integrity. Constraints and requirements for software design, testing and modification will be associated with each safety integrity level.

However, safety integrity is not only a matter of software design. During software transportation, installation, and operation, some activities can be necessary to achieve a sufficient safety level. Activities such as granting of user access rights to users based on allowed software functionalities or data areas, and transporting critical or secure data properly are important and relevant to safety.

6.11 Security

Security is an important aspect with respect to software. There are several methods to meet security requirements for software usage, such as:

- cyclic redundancy check
- encryption
- digital signature
- activation or license keys

The security classification of data, executable code, and documentation can constrain software support activities. The main influence on equipment will be to impose special handling requirements. This can limit access to the software and introduce design requirements that emphasize the importance of specific software support tasks and equipment. The security classification of a software item will depend on its application and equipment design. Wherever possible, software design must ensure that highly classified software parts are physically segregated from all other software within the Product. Security requirements provide criteria for security classification of software items and specify modification and handling constraints associated with such classifications.

Another aspect of security relates to operation. It is necessary to ensure that a software package can operate in a secure environment. This means internal security concerning a proper control policy for user access, as well as external security. Therefore, security must effectively block intruders from outside, and limit and control the activities of internal users (eg, via the internet). These operational security requirements can cause several support activities, such as:

- network administration activities (eg, granting of user access rights to specific data areas, file servers or application services)
- firewall administration activities
- virus detection and defense activities

Note

Software in defense Products can contain operational data that third parties must not access. In this case, it must be possible for the crew to erase all software and data as fast as possible, to prevent further use of the Product and/or access to the data.

6.12 Size

The size of software packages influences supportability parameters, in terms of expected level of change traffic and resources required to implement modifications. The size of the software depends on the application and the design solution. Software requirements must state any constraints on the size of run-time software as imposed by the software design. Many software support and supportability projections will be based on software size and complexity. Software development requirements must define aspects for data collection and analysis to measure software size and to verify models or estimates of supportability parameters, which are dependent on software size.

6.13 Competence

It is necessary to consider the required competence for personnel involved in software-related activities from several perspectives. The simple user must be able to handle the functions of the

software in a professional manner. Operators with administrative responsibilities will require a deeper education concerning the functionality of the software in the corresponding IT environment. Software modification requires the highest level of competence. In this case, personnel must have appropriate software engineering knowledge. Requirements for qualification are associated with the application domain, and the technology or methods used. The hardware and software design, as well as the applicable software support policy will determine competence requirements.

6.14 Software distribution

It is possible to distribute software on a variety of media, for example the internet or other network solutions, such as Local Area Network (LAN) or Wide Area Network (WAN) structures.

If it is necessary to transport software using hardware devices, the type of storage medium and access restrictions to that medium must be taken into account.

6.15 Standardization

It is possible to apply standardization to the computing environment within which the software is operated. The same applies to technologies and engineering processes used to develop the software and the associated software documentation. Standardization helps to reduce the diversity of tools and methods. This considerably reduces the effort for personnel training, and the costs for the required equipment at support facilities. Standardization can be helpful for operational aspects as well (eg, standardization of loading/unloading or backup procedures to conform to existing standards).

6.16 Technology

Software engineering methods and tools used in development and implementation must consider some technology aspects, including:

- software design methods and supporting tools
- operating systems
- programming languages
- compilers and debuggers
- software test methods and test environments (hardware and software)
- project specific tools and methods

Technology also influences operational aspects. For example, the technology of storage devices can significantly influence support activities, and the use of network-based technologies can dramatically ease the maintenance of client computers in a client-server application environment.

7 Associated parts of the S3000L data model

The documentation of the associated data to this chapter is supported by the following Units of Functionality (UoF), refer to Chap 19:

- S3000L UoF Applicability Statement
- S3000L UoF LSA Candidate
- S3000L UoF Part Definition
- S3000L UoF Task Requirement
- S3000L UoF Task Usage
- S3000L UoF Time Limit

Chapter 14

Life cycle cost considerations

Table of contents

List of tables

List of figures

References

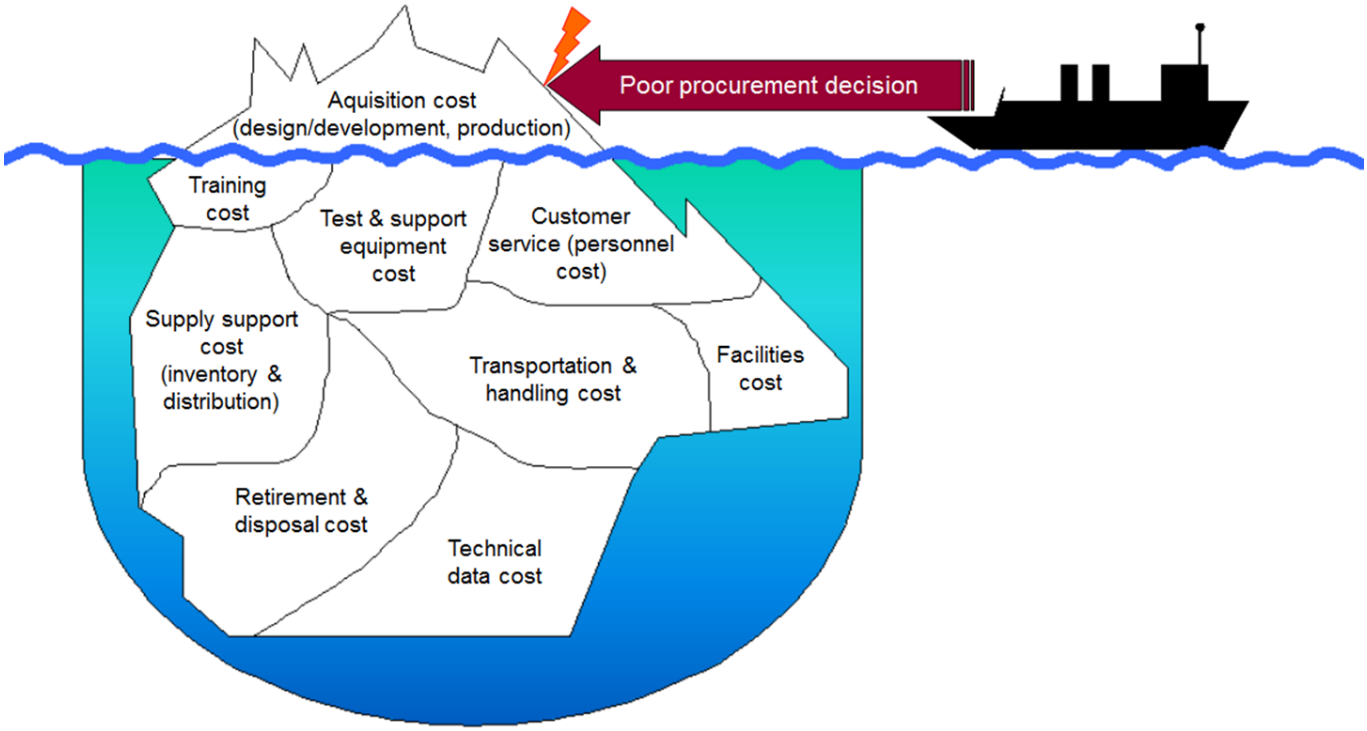
Table 1 References

Chap No./Document No.	Title
Chap 19	Data model
NATO RTO Technical Report 028	Cost Structure and Life Cycle Costs for Military Systems

1 General

1.1 Introduction

In most cases, Operating and Support (O&S) costs significantly exceed all costs of acquisition. Therefore, the Product's initial cost (acquisition cost), its O&S costs, and disposal costs should influence the decision to purchase a Product. Fig 1 indicates the relation between acquisition cost and support-related cost for buying decision.



ICN-B6865-S3000L0070-002-01

Fig 1 Acquisition costs versus O&S costs (iceberg effect)

LSA identifies the support system required to meet the customer's performance objectives in terms of product usage (eg, availability, safety). In that context, LSA requirements support the minimum LCC by meeting requirements in areas such as the number of maintenance levels, staff involved in maintenance support, maintenance training limitations, technical manuals limitations, reliability, and Mean Time To Repair (MTTR) at each maintenance level. The LCC analysis process can be used to estimate the cost impact of LSA requirements. Therefore, LCC analysis results can be used as decision criteria in the LSA process to:

- select Candidate Items (CI)
- identify analysis activities for each CI
- influence the design
- assess the configuration

- perform Level of Repair Analysis (LORA)

To the same extent, the LCC analysis process benefits from LSA as the data generated for the related support requirements are an input for the LCC model. Therefore, it is vital to maintain a coherency between the LSA results and the Data and Assumptions Definitions Document (DADD) used for the LCC analysis. Refer to Para 3.3.

1.2 Purpose

The purpose of this chapter is to provide LSA focal points and analysts an understanding of LCC, its use and interactions with the LSA business process. LSA focal points and analysts must share information with LCC experts on a regular basis, to ensure that their analysis and decisions take into consideration the economical factor.

1.3 Scope

This chapter provides an overview of the LCC aspects and indicates which information developed by LSA processes can be used to support LCC analysis activities.

2 Different views on costs over the Product life cycle

Several terms are used to define the cumulative costs of a Product and its support environment over its life cycle from design to disposal:

- LCC
- Total Cost of Ownership (TCO)
- Whole Life Cost (WLC)

These terms have different scopes (refer to NATO RTO Technical Report 028), described in Para 2.1, Para 2.2 and Para 2.3 respectively.

For LCC analysis activities, there is a relationship to LSA activities and results. Refer to Para 4.

2.1 Life cycle cost

LCC consists of all the direct costs plus any indirect-variable cost associated with the procurement, operations, support and disposal of the Product. Costs depending on indirect variables include linked costs, such as additional common support equipment, additional administrative personnel and non-linked costs, such as new recruiters to recruit additional personnel. LCC also includes marginal costs (both direct and indirect) for new equipment, configuration, or capability.

LCC does not include notional allocation of costs, whereas TCO and WLC can. In addition to that, LCC does not include any indirect-variable costs related to activities or resources not affected by the introduction of the Product.

Note

LCC is used as the starting point to analyze alternatives and to compare options of alternatives, and often for economic analyses.

2.2 Total cost of ownership

TCO consists of all elements that are part of LCC plus the indirect, fixed and linked costs. These can include items such as common support equipment, common facilities, personnel required for unit command, administration, supervision, operations planning and control, fuel and munitions handling.

TCO represents all costs associated with the ownership of a system, except non-linked fixed costs related to managing the organization.

Note

TCO is used for budgeting and optimization purposes, for the definition of services used in different systems, and for financial analysis.

2.3 Whole life cost

WLC consists of all elements that are part of TCO plus indirect, fixed, non-linked costs. These can include items such as family housing, medical services, ceremonial units, basic training, headquarters and staff, training academies and recruiters.

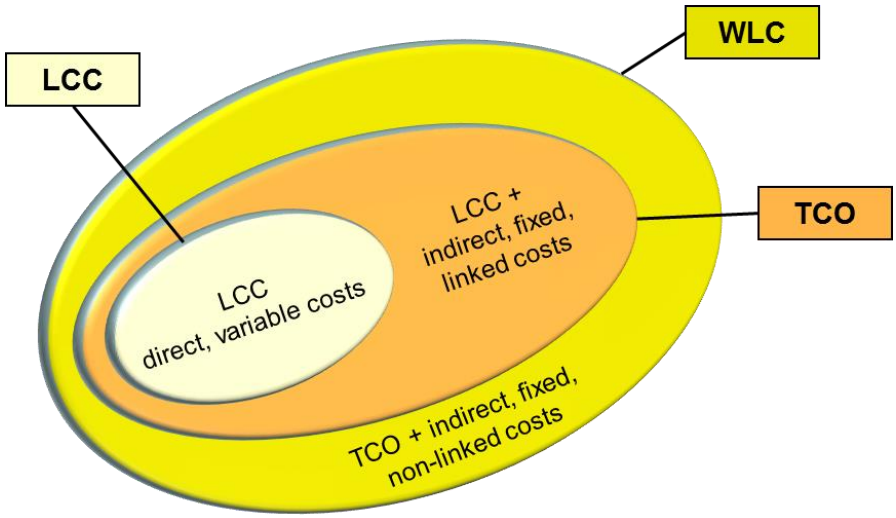
In WLC, all costs or expenses borne by the organization are linked to the relevant Products.

As WLC represents the total budget provision including elements such as headquarters costs, it provides an overview of the allocation of all funds.

Note

WLC is used for a strategic view and high-level studies.

Fig 2 shows the relationship between the different cost classes described in Para 2.1, Para 2.2 and Para 2.3.



ICN-B6865-S3000L0071-003-01

Fig 2 Relation between LCC, TCO and WLC

3 Life cycle cost analysis process

3.1 General approach

Although some changes to the LCC analysis process can be necessary based on the Product, all LCC analysis processes have some essential steps in common. The depth of the analysis for every step needs to be tailored to the requirements of each project.

The approach varies depending on the issues to address, the cost requirements, and the availability of suitable data. With some variation to the details, it is possible to apply the same LCC analysis approach to all projects, regardless of their specific requirements. This approach requires the following steps to achieve a successful LCC analysis:

- definition of aims and objectives of the study. Refer to Para 3.2.
- development of the LCC framework. Refer to Para 3.3.
  - definition of the program content and budget within the LCC breakdown structure
  - identification of methods and models for the objective and the life cycle phase
  - identification, collection and documentation of data and assumptions necessary to populate the LCC model
  - analysis of risks and uncertainties
- report and presentation of the results. Refer to Para 3.4.

3.2 Aim and objective of the study

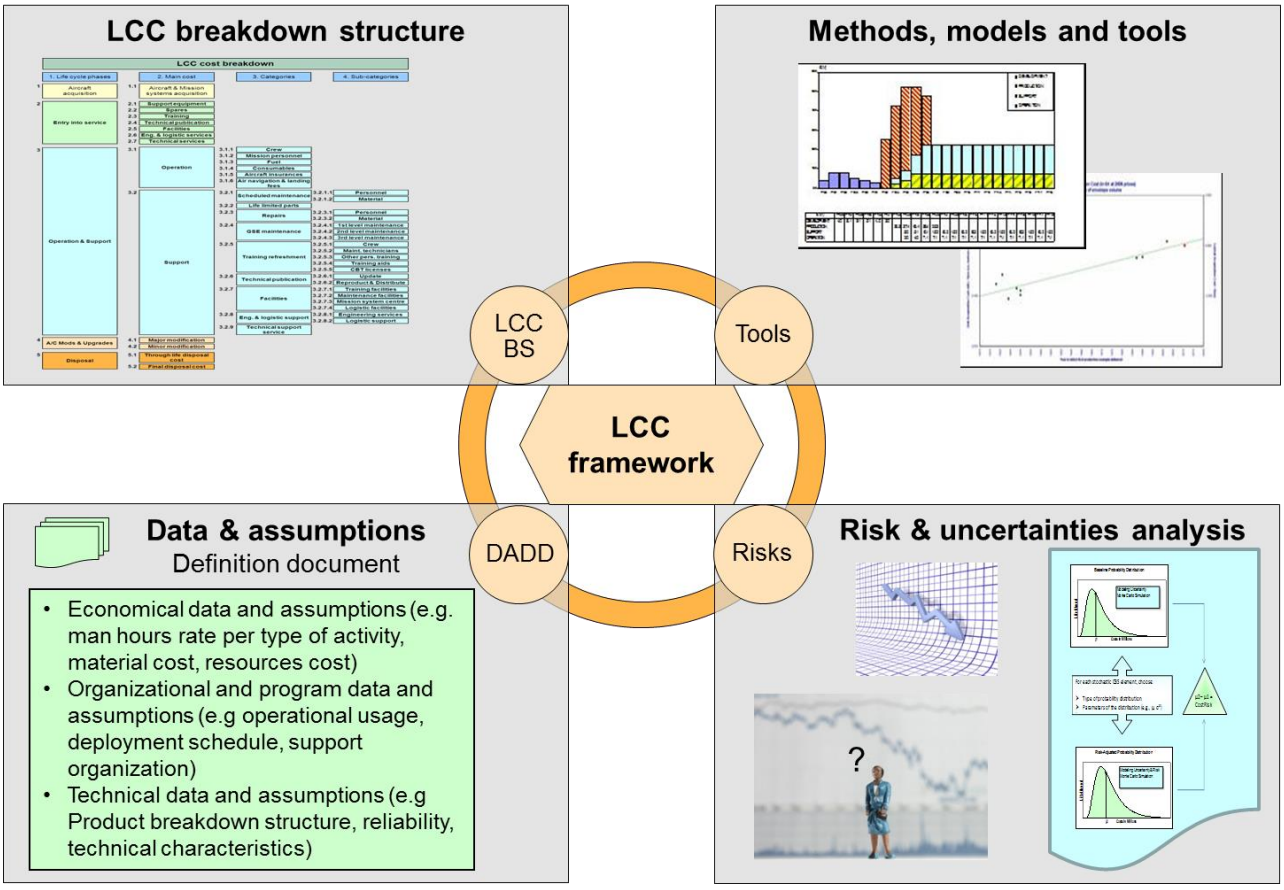
A clear definition of the object of study is essential to understand what the estimates will be used for (eg, setting budgets, support options evaluation, trade-off decisions, pricing). Examples of the range and variety of objectives are:

- the acquisition of a new transport aircraft to fill a capability gap
- the acquisition of an information management system for a medium-sized company
- the modification of a large fleet of vehicles due to obsolescence

The aim and the objectives of the study will have a major impact how an LCC analysis is conducted. A different type of question will result in a different way of conducting the study. This will often be implicit in the type of study, nevertheless it is necessary to define aim and objective of LCC studies clearly and unambiguously to provide useful and meaningful results.

3.3 Develop the life cycle cost framework

Fig 3 shows the different components of an LCC framework.



ICN-B6865-S3000L0072-002-01

Fig 3 Life cycle cost framework

3.3.1 Life cycle cost breakdown structure

The LCC Breakdown Structure (LCCBS) identifies all cost items affecting the total LCC of the Product. It is necessary to define cost items in a specific way to avoid ambiguity. For that purpose, there are five dimensions that help when defining a cost item:

- resources
- activity
- product

- time
- location

Example:

Cost of technicians (resources) to maintain (activity) the engine (Product) for 25 years (time) at company X (location).

3.3.2 Data and assumptions definitions document

The Data and Assumptions Definitions Document (DADD) records the LCC analysis ground rules and assumptions. This includes organizational, program, technical and economic data. The LSA can provide technical and organizational data regarding support. For this purpose, it is necessary to keep and maintain LSA data during the in-service phase to enable LCC analysts to use the LSA data as a dependable information source.

3.3.3 Methods, models and tools

There are many software packages for LCC analysis, but it is recommended to select the most appropriate tools depending on:

- the specific Product to be analyzed
- the objectives of the LCC analysis
- the life cycle phase

Given the availability of several methods, it is useful to compile all results in an aggregation tool to maintain an overview of the LCC. In particular, government customers often use a single standard software to manage every kind of national project. Contractor and customer tools must synchronize their LCC analyses. In some cases, tailored LCC models are more advisable.

3.3.4 Risk and uncertainties analysis

Risk and uncertainties analysis can be included within the model to provide a clear illustration of possible cost variations in the program estimates.

3.4 LCC analysis outputs and reports

A variety of reports and graphs is available to illustrate the results of the LCC analysis and provide a clear explanation to users/customers on the analysis outcomes and implications.

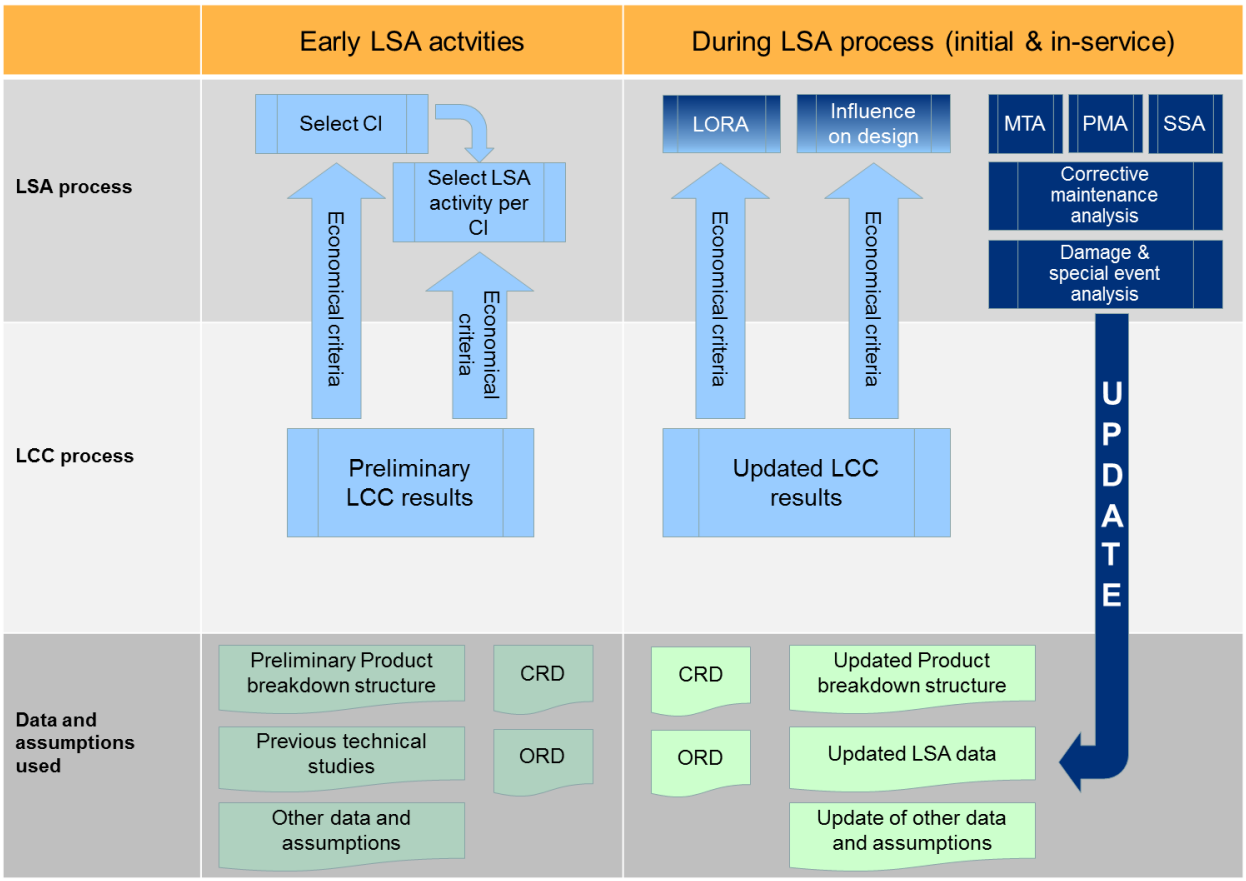
The final LCC analysis report must contain at least the purpose and scope, a summary of the LCC framework, the results and their analysis, including sensitivity and trade-off analysis, conclusions and recommendations.

4 Life cycle cost analysis in the LSA process

4.1 Interactions

LCC analysis work starts in the early phases, before LSA begins. Therefore, LCC results are already available once the LSA activities start. LCC results can serve as criteria to set up the LSA program plan and framework, support the selection of LSA CI, and the identification of analysis activities for each LSA candidate. Refer to Fig 4.

During the LSA process, the different types of analysis, for example corrective maintenance analysis, Preventive Maintenance Analysis (PMA), damage and special events analysis, Maintenance Task Analysis (MTA), or Software Support Analysis (SSA) will generate a large amount of data. These inputs are vital for the update of the DADD of the LCC framework, particularly with regard to resources. At the same time, LCC analysis also serves as critera to make decisions during activities such as influence on design, configuration assessment, and LORA.



ICN-B6865-S3000L0073-002-01

Fig 4 Interactions between LCC and LSA

4.2 Life cycle cost aspects during early LSA activities

Before LSA work starts, there are many reasons to provide LCC analysis, such as:

- affordability analysis
- business case
- evaluation of alternatives

Those analyses include some useful data and assumptions, for example:

- Organizational and program data, such as support and maintenance organization, operation and the quantity of the Product. ORD and CRD usually contain these data.
- Technical data, such as Product breakdown structure, including physical, functional or any other kind of Product breakdown structure as applicable, maintenance and support characteristics of the breakdown elements based on existing analysis results (eg, weight, reliability and testability information, duration of maintenance, periodicity of preventive maintenance, scrap rate).
- Economic data (eg, man-hour costs for different activities like design, development, production, operation or maintenance, costs of other resources)

At this early stage, LCC models are usually based on parametric or analogy methods, with a corresponding degree of uncertainty. However, the LCC model developed identifies cost drivers based on the breakdown elements and the corresponding support activities. This identification serves as one decision criterion during the LSA preparation for CI and analysis selections, such as:

- LCC analysis can support the selection of LSA CI from the Product breakdown that are cost drivers and have many uncertainties
- For each CI selected, LCC analysis can identify the activity driver. For example, if corrective maintenance is the activity driver, it is advisable to assess in-depth technical Failure Modes and Effects Analysis (FMEA) results and perform MTA.

4.3 Life cycle cost analysis during LSA process

During the LSA process, it is possible to update the LCC analysis with the results of different analysis activities. It is vital to maintain a coherency between LSA data and the DADD used for the LCC analysis.

LCC analysis also serves as a decision criterion for other activities, such as influence on design, configuration assessment, and LORA.

4.3.1 Influence on design

For influence on design, the following steps can help in the decision-making process. However, it is recommended to use these steps in combination with other criteria, such as safety and availability. The steps are:

- use the LCC model to identify breakdown element drivers (high LCC)
- perform LSA to decrease the LCC support cost for these drivers and to identify LSA requirements
- analyze whether these requirements justify a design change
- analyze all necessary changes together with the design department, collect data and assumptions for design, development, production and disposal activities
- update LCC analysis taking into consideration new design, including LSA requirements
- compare with initial LCC to quantify benefits

4.3.2 Configuration assessment

If any other reason, other than those stated at Para 4.3.1, requires a design change, LCC analysis serves as decision criteria during the configuration assessment. Certain steps can help in the decision-making process. However, these steps need to be combined with other criteria such as safety and availability. The steps are:

- identify the data and assumptions in the change request
- perform LSA and identify LSA requirements if the change concerns CI
- update LCC analysis with all the data above

4.3.3 Level of repair analysis

LORA uses LCC analysis to evaluate alternatives in terms of levels of repair from an economical perspective. LCC aspects that are taken into account are maintenance, spares provisioning, training maintainers, support equipment and facilities, non-recurring and recurring costs.

5 Associated parts of the S3000L data model

The documentation of the associated data to this chapter is supported by the following Units of Functionality (UoF), refer to Chap 19:

- S3000L UoF Product and Project

Conversely, the LSA process generates data that are useful for LCC analysis. Corresponding LSA data are included in the following UoF:

- S3000L UoF Breakdown Structure
- S3000L UoF Damage Definition
- S3000L UoF LSA Candidate
- S3000L UoF LSA Failure Mode

- S3000L UoF Part Definition
- S3000L UoF Performance Parameter
- S3000L UoF Product Usage Context
- S3000L UoF Special Event
- S3000L UoF Task
- S3000L UoF Task Resources
- S3000L UoF Task Usage

Chapter 15

Obsolescence analysis

Table of contents

List of tables

List of figures

References

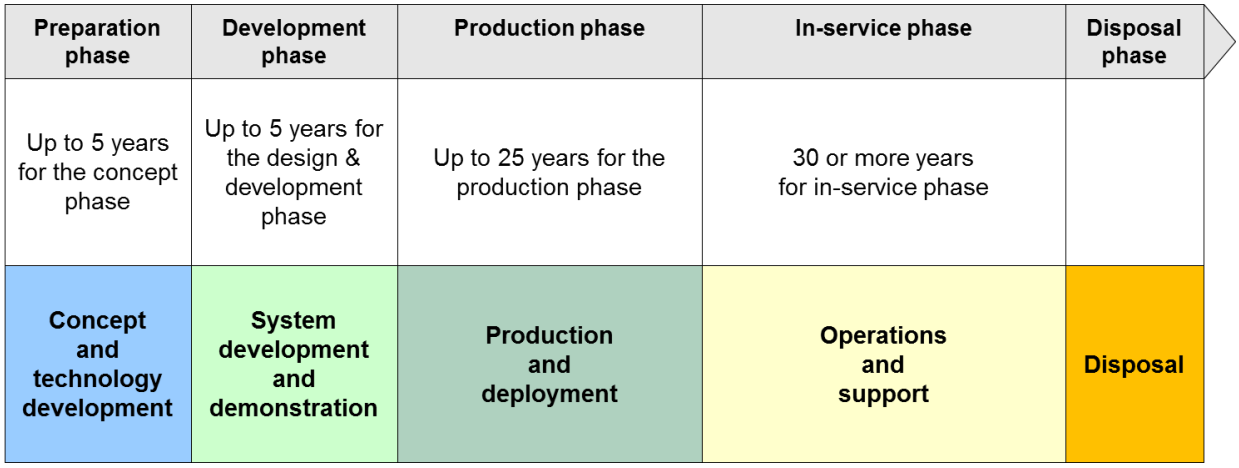
Table 1 References

Chap No./Document No.	Title
Chap 19	Data model
EN 62402:2007	Obsolescence management - Application guide

1 General

1.1 Introduction

Obsolescence occurs due to the length of time it takes to develop and launch a Product and their subsequent long life cycles (refer to Fig 1). Obsolescence affects all Products and systems, hardware and components, as well as test and support equipment, software, tools, processes, IPS elements, standards, specifications, and expertise.



ICN-C0419-S3000L0093-003-01

Fig 1 Life cycle phases

Note

In many cases, the different life cycle phases overlap.

Obsolescence occurs for a number of reasons:

- The life span of Product components decrease, especially in the case of electronic components (approximately 5 years or less per innovation cycle)
- The manufacturing base (eg, subcontractors and vendors) are subject to market forces. Manufacturers can go out of business and essential parts or sub-assemblies can become unavailable. This phenomenon is also known as Diminishing Manufacturing Sources and Material Shortages (DMSMS) or Diminishing Manufacturing Sources (DMS). More specifically, DMSMS and DMS indicate the loss or impending loss of manufacturers or suppliers of items or raw materials. DMSMS/DMS and obsolescence are often used interchangeably. However, obsolescence refers to a lack of availability due to statutory or process changes and new designs, whereas DMSMS is a lack of sources or materials.
- The loss of design and technical know-how can have a big impact on long-life cycle Product supportability, particularly if bespoke software and components were used in the original design
- The pace of obsolescence has increased due to the ever-growing number of environmental regulations. Those regulations determine specific chemicals (eg, Registration, Evaluation, Authorization and Restriction of Chemicals known as REACH) or materials (eg, beryllium, copper) can be used. It is recommended that this aspect be considered from the earliest phases of a project.
- As a result of the costs of scarce materials, production facilities and support services (eg, software maintenance), Product support can become unaffordable

The pace of technological innovation, coupled with prolonged use of Products and services, makes it almost inevitable for the obsolescence to impact the Product life cycle at some point. Despite this, it is possible to manage and mitigate obsolescence. However, the cost of

mitigation increases significantly as the system, Product, or service moves closer to becoming obsolete. Therefore, it is recommended that Obsolescence Management (OM) be undertaken as early as possible. OM must be an integral part of the preparation, development, production and in-service support phases to minimize potential remedial expenditure and overall Life Cycle Cost (LCC).

1.2 Purpose

The relationship between LSA and obsolescence is that both processes occur repetitively in the Product life cycle. Obsolescence must be an integral part of the LSA process, as it affects supportability, operability and LCC. Therefore, it is necessary to manage obsolescence and LSA closely until the disposal phase. This chapter provides a description of obsolescence analysis and the implementation of the associated activities. It offers guidance on obsolescence planning to mitigate the risk of obsolescence and minimize the LCC of long-life cycle Products.

Note

The information in this chapter gives an introduction and an overview of the issues related to OM. Although only applicable to software, the European standard EN 62402:2007, Obsolescence management - Application guide, provides authoritative guidance on the subject.

1.3 Scope

The target readers of this chapter are engineering, quality, support and supply chain organizations. It provides guidance on:

- defining an obsolescence strategy
- implementing obsolescence solutions
- developing an OM plan
- linking obsolescence to the risk management activity
- obsolescence and LCC
- ensuring OM costs are budgeted for
- ensuring industry plans and manages obsolescence
- identifying necessary data

2 Obsolescence strategy

Reactive and proactive strategies are the basic approach to managing obsolescence, and are outlined below.

2.1 Reactive obsolescence management

Reactive OM implies that there is either no specific provision to manage obsolescence or solve obsolescence issues when they occur. Unless occurring at the same time, each issue is handled separately, and there is no link with the risk management activity. The budget for issue resolution either comes from budget allocated to other activities, or from additional funds. Table 2 lists all reactive management options:

Table 2 Options for reactive obsolescence management

Option	Description
Do nothing	This is a valid choice. The decision could be based on the extended life and very low consumption of the item. This obviates the need for additional procurement activity.
Re-design	Usually, an expensive option but it can be unavoidable. Limited design transparency can mitigate costs.

Option	Description
Alternative part procurement	This activity usually results in a search for spare parts, including the removal of serviceable parts from an unserviceable Product to repair another unserviceable Product and make it serviceable. This is known as cannibalization.

2.2 Proactive obsolescence management

Proactive OM implies specific resource provision and a maintenance plan for obsolescence issues. It is part of the risk management activities and budget is allocated to it. A group of identified stakeholders address the issues as they arise. Product monitoring creates time to review resolution options and justify decisions. It is recommended that all programs consider the proactive strategy at their start. Table 3 lists all proactive OM options:

Table 3 Options for proactive obsolescence management

Option	Description
Do nothing	As for reactive obsolescence management
Lifetime or end of life buy	A valid approach, which can offer the best value. However, it is necessary to consider possible equipment life extension, changes in consumption rates and costs of ownership (eg, storage, handling, transportation).
Planned upgrades	With planned upgrades, multiple issues can be addressed at the same time. However, they must be coupled to obsolescence monitoring activities whether the upgrade is required or due to obsolescence. It is also necessary to ensure sufficient spares are available between upgrades and budgets strictly controlled.
Obsolescence monitoring	A key component of any OM strategy. If implemented correctly, it will at least ensure that enough time is available for planning and finding a solution. Once the obsolescence strategy is established, it is possible to select the preferred solution from the available options.
Design out obsolescence	This approach needs to be implemented in the earliest phases of a project. The intention is to design the equipment so that a subassembly capability is not linked to a particular solution.

2.3 Combination of reactive and proactive obsolescence management

A combination of reactive and proactive OM proves to be the most suitable approach for some projects.

This can be particularly appropriate when there is a large number of components, some of which are identified as low risk. This option reduces the management cost of the OM task and focuses on high-risk items. However, this approach entails an increased risk. In proactive OM, costs can increase with the number of components considered and monitored. It is necessary to weigh these costs against the possible impact on availability and obsolescence rectification costs. The cost of OM influences the choice of the strategy.

It is necessary to evaluate thoroughly the use of reactive OM in the early phases of a project. In fact, it can be impossible to recover important information to support proactive OM later in the project.

2.4 Obsolescence monitoring

There are various options available to implement the obsolescence monitoring solution. The use of the Design Authority (DA) to manage obsolescence is an obvious choice. The DA is the best choice to develop a list of parts, identify solutions and modify the design plan. This is particularly the case during the early phases of the project, when obsolescence analysis can influence the design. During the in-service phase in particular, an alternative solution is the use of a third-party contractor with a dedicated OM portfolio of skills. Many companies offer an OM service. The advantages of this approach are the access to specialized skill sets and experience in the implementation of OM. Overheads can cover many other projects, and therefore offer better value. These companies do not have a vested interest in identifying obsolescence issues unless they are also contracted to identify solutions. A further option would be to employ a combination of DA and third parties. DA can take care of the bulk of the work, while third parties can offer specialist skills and, because of their independence, verify DA obsolescence claims, as the DA can have a vested interest in such claims.

2.5 Strategy tools

Impact, probability and cost are essential elements in the decision-making process for OM strategies. At first, the data supporting this process can be based solely on the judgment and experience of the team, but this must be improved as the project progresses. The process considers:

- The **impact** of an event on the capability of the system. It is necessary to evaluate it with the customer or user, including all reliability, maintainability and availability aspects.
- The **probability** of the event occurring, including possible environmental and safety legislation aspects
- The **cost** of an obsolescence event. This includes the total Product life cycle cost addressing the design, production and implementation elements of the system and any re-work or support activity costs (total ownership cost).

These three elements are considered together to determine the classification of the obsolescence event. This classification is useful to determine the obsolescence strategy, the level of monitoring, the need for further investigation, and the periodicity of review.

3 Develop an obsolescence management plan

The first step in managing obsolescence is to formulate and document a comprehensive OM plan. This document will implement the relevant strategy for the life of the Product or project. It is recommended that this be developed at the earliest phase of the project and reviewed at least at the end of each phase for applicability. It often starts as quite a small document but will develop and be refined as the lessons and decisions made during the design and development phases are incorporated. As the project progresses, the plan will develop into a through-life record of the strategies employed, options considered, and decisions made. It can incorporate the contractor's OM plan, to be provided in response to the invitation to tender and contract.

The OM plan must:

- achieve the optimum compromise between life cycle cost, performance, availability and maintainability for the entire Product
- cover all material, including hardware, software, tools, test equipment, human resources and training
- be compatible with the customer’s current support arrangements
- provide a clear basis for obsolescence management requirement negotiation with suppliers and partners
- take into account the need for component or equipment requalification/re-certification following component or module substitution
- identify the linkages and interfaces to the risk and life cycle cost program activities

The OM project team defines the scope and content of the OM plan, which must reflect the scope, complexity and cost of the project. The OM plan must contain at least the aspects covered by Para 3.1 through Para 3.8.

3.1 Obsolescence specifications

The obsolescence planning document is used to identify obsolescence issues in the Product, record the chosen strategy, provide details of plans and OM decisions, document the reasons why options were considered, analyses carried out and the trade-off decisions made. It is recommended that subsidiary documentation contains a full record of the details.

3.2 Budget and life-cycle cost for obsolescence management

It is essential that budgetary provision be made for the OM task itself. OM carries its own resource cost, that must be included in the project budget. In addition, costs for obsolescence issues must be included as issues arise. It is necessary to review the costs regularly since the issues will vary as the project progresses. Cost forecasting must be linked to the project risk activity to ensure a consolidated approach and avoid items are considered twice or not considered at all.

Obsolescence can lead to major changes in design, support concept and organization. As obsolescence can impact all phases of a project, any option review must consider the implications for the Product life. Therefore, OM must be linked to the LCC activities. The correct approach to identify cost issues to be included in the appropriate management plans is an approach that takes into consideration the complete Product life cycle.

3.3 Linking obsolescence to the risk management activity

This part considers the operational risks associated with obsolescence over the life of the equipment:

- What would be the impact of the Product being unavailable due to lack of spares or degraded performance due to substituted parts?
- What would be the likely cost of premature replacement or of other measures to circumvent obsolescence?
- What is the probability of obsolescence occurring (including considerations on technology advancement and new legislations)?

Obsolescence is a key contributor to any risk assessment. It is recommended that the strategy for addressing obsolescence in risk processes be established as early as possible in the project and be reflected in the appropriate management plans. Having good visibility of the obsolescence risks is the first essential step to minimize negative effects. This way, the project management team will identify and implement the most appropriate action to deal with issues in the most cost-effective way. It is essential to acknowledge that the Product user is affected the most during the in-service phase in terms of readiness, supportability, and life cycle costs. Obsolescence is always a risk, but it can create opportunities when alternative parts offer enhanced performance.

3.4 Stakeholder identification

This part is used to record the details of the stakeholder community personnel authorized to take decisions on trade-offs between cost and impact. OM requires input from customers, OEM, subcontractors and, in some cases, suppliers. It is necessary to form a team with the appropriate stakeholders. Team members will vary based on the Product and potentially the Product life cycle. The team must establish a good working relationship to gather information and take decisions quickly. The longer it takes to implement a solution, the less chance of it having success.

3.5 Scheduling

This part is used to arrange a schedule and strategy to renew Product components and allow capability upgrades, including future Product support. The strategy will depend on the Product and can be broken down into electrical/electronic parts, systems or subsystems and non-electrical/electronic parts, systems or subsystems. Scheduling also determines how often the scanning of the Product breakdown must occur. If scanning occurs frequently, an obsolescence alert will less likely become a serious issue. However, performing this activity more often will result in a more expensive process.

3.6 Standards and tools

This part is used to identify obsolescence standards and to set a tool used to predict the life cycle of electronic components, enable a review of bill of materials and predict when support for an electronic component will no longer be available. The tool must be able to read all levels of a Product breakdown and to predict when a component or assembly can become obsolete.

3.7 Supply chain

This part is used to maintain a record of DMSMS or DMS notices received from suppliers. DMSMS or DMS notices alert customers that production is either stopped or limited for a specific part. The notices usually contain part numbers, last order and shipment dates, and minimum order quantities. In OM, it is essential that DMSMS or DMS notices be collected proactively.

3.8 Obsolescence data

This part defines a secure data repository for obsolescence data. It is essential to establish a secure area to exchange and store information. This protects the intellectual property of all parties. These data can be used again for similar Products at the same stage of their life cycle.

4 Implementation of obsolescence solutions

When obsolescence occurs, the chosen solution depends completely on the project and market circumstances at that time. Therefore, it is necessary to treat each occurrence an individual event and consider at least these aspects:

- reclamation
- last time buys
- lifetime buys
- re-manufacture
- use of an interchangeable item
- adaptive solutions
- reverse engineering
- new design

5 Ensuring obsolescence planning and management

Industry must develop a plan for obsolescence as early as the beginning of the design process. Supportability engineering and logistics activities must include “obsolescence prevention” among their objectives. This objective covers all the phases of the project. Industry must:

- produce an OM plan
- conduct obsolescence critical analysis in the project, to identify the Product items with the highest risk of obsolescence and, based on some predefined criteria, use similar methodologies for safety hazard and mission critical analyses
- adopt a pragmatic approach by putting into practice, as a continuous effort with all levels of industry participants, timely and cost-effective obsolescence engineering practices, management tools, methods and practices to avoid the negative effects of obsolescence on the project

- prepare and include obsolescence risk mitigation strategies in the Statement Of Work (SOW) for all provisioning contracts during the Product life cycle
- detail OM data requirements clearly within the SOW and discuss rights of access during commercial negotiations
- determine the obsolescence status of parts before their procurement
- ensure that all stakeholders plan the most cost-effective time scales for in-service technology updates/upgrades or technology road maps, and manage obsolescence in existing equipment to extend its service life in line with the logistics and necessary qualification/certification processes
- apply practices already established in other projects for obsolescence mitigation whenever deemed convenient and cost-effective

It is essential that the invitation to tender include the requirement for an OM plan for all project phases. Contractor's best practices can provide the format and contents of the plan. However, the plan must at least conform to the requirements of a recognized standard. It is necessary to consider intellectual property rights with regard to OM, to ensure access to required data to support the OM strategy. The requirements of OM must also be considered in the context of an exit strategy, to ensure that all the data needed to manage obsolescence are available in the event of a contract termination.

Note

One commercial strategy is to place full responsibility for obsolescence management to the contractor. The simplest form of this hands-off solution consists of a fixed-price payment to solve any future obsolescence issue. This solution is not optimal from a technical point of view, but rather a more commercial or contractual approach, as it passes all liability for future obsolescence issues to industry during a specified period. In some cases, a redistribution of the payment plan may occur by bringing forward some payments to cover potential obsolescence issues arising during a specified period. However, this approach has some drawbacks. Although a fee is paid to "transfer the risk to the contractor", care must be taken to ensure that, at a minimum, there is limited visibility from the customer side so that he can see where action is being deferred or risk carried. These decisions can affect the customer's decision on whether to close the contract. Alternatively, it is recommended that provision be made to address obsolescence risks towards contract completion, if applicable. In addition, during the fixed-price negotiation, it can be difficult to justify or investigate costs unless the contractor proposal includes a detailed risk-based obsolescence cost analysis.

6 Associated parts of the S3000L data model

The documentation of the associated data to this chapter is supported by the following Units of Functionality (UoF), refer to Chap 19:

- S3000L UoF Part Definition

Chapter 16

Disposal

Table of contents

List of tables

List of figures

References

Table 1 References

Chap No./Document No.	Title
Chap 19	Data model
Chap 22	Data element list
European Community Directive 2000/53/EC	Directive on End-of Life Vehicles (ELV)
European Community Directive 2002/95/EC	Restriction of The Use of Certain Hazardous Substances in Electrical and Electronic Equipment (RoHS)
European Community Directive 2002/96/EC	Waste Electrical and Electronic Equipment (WEEE)
European Community Directive 2006/121/EC	Registration, Evaluation, Authorization and Restriction of Chemicals (REACH)
European Union Directive 67/548/EEC	Directive on the approximation of laws, regulations and administrative provisions relating to the classification, packaging and labelling of dangerous substances
Regulation (EU) No 517/2014	Fluorinated Greenhouse Gases and Repealing (F-Gas)
Regulation (EU) No 850/2004	Persistent Organic Pollutants (POP)
Regulation (EU) No 528/2012	Biocidal Products (BPR)
Regulation (EC) No 1907/2006, article 31 and ANNEX II	Requirements for the Compilation of Safety Data Sheets (SDS)
IAEA RS-G-1.8	Environmental and Source Monitoring for Purposes of Radiation Protection Safety Guide
IAEA WS-G-2.7	Management of Waste from the Use of Radioactive Material in Medicine, Industry, Agriculture, Research and Education Safety Guide

Chap No./Document No.	Title
IAEA INFCIRC/386	Code of Practice on the International Transboundary Movement of Radioactive Waste
NR528 DT R03 E	Green Passport (Bureau Veritas)
IAEA SSR-5	Disposal of Radioactive Waste (2011)
IMO Resolution A.962(23)	Guidelines on Ship Recycling
Resolution A.980(24)	Amendments to the IMO Guidelines on Ship Recycling (Resolution A.962(23))
STANAG 4518 Edition 1 (2001)	Safe Disposal of Munitions, Design Principles and Requirements, and Safety Assessment

1 General

1.1 Introduction

Disposal is both a business process and the name of a phase in a Product life cycle. This chapter describes the disposal business process.

The establishment of a procedure for disposal of hardware or software is required to ensure that appropriate care is taken to dismantle and dispose of an individual Product (at the end of its life cycle) and its equipment, components, structural parts and consumables in the context of Product usage or maintenance during the whole in-service phase. It is necessary to perform dismantling and disposal in a safe, sustainable, and environment-friendly manner.

Disposable items include hardware used for recycling purposes and to avert air, water or soil pollution, as well as unsafe handling and dumping of hazardous materials, but also software used to store or delete classified, confidential, or proprietary software or data.

For the purpose of an S3000L implementation, it is necessary to consider:

- that disposal occurs only once for a complete Product during a Product life cycle. Once disposed, the Product no longer exists
- it is necessary to consider disposal constantly for all Product components, consumables or support equipment that are no longer usable because they are broken, damaged, expired or at the end of their lifetime, as applicable
- the probability that disposal will occur is 100%, at some point and in some manner

As a consequence, design must implement disposal requirements from the very beginning of a development program. The general disposal principles to be applied are:

- to recycle as much material as possible
- to minimize landfill waste resulting from the Product's life cycle

Therefore, it is necessary to consider disposal as an important business process implemented within the framework of an IPS/LSA program development strategy.

Disposing of a Product means phase it out at the end of the Product's service life. Typically, the disposal of Products occurs during the last phase of their life cycle. However, it is necessary to consider disposal from the earliest phases of every project that acquires a Product.

1.2 Purpose

This chapter provides guidance for developing disposal procedures that are proved to be safe, efficient, environmentally acceptable, and cost effective.

It also provides guidance on required operations and disposal tasks for the Product to:

- identify support resource requirements for each task
- identify new or critical support resource requirements
- identify transportability requirements
- identify support requirements that exceed established goals, thresholds or constraints
- provide data to support participation in the development of design alternatives to reduce disposal costs and optimize support resource requirements
- provide guidelines on the actions needed during each project phase and the methods to measure the effort properly
- provide source data for the preparation of required IPS documents (eg, disposal instructions, training program, manpower and personnel lists)

1.3 Scope

Disposal for hardware must consider the following aspects:

- the destruction/neutralization of toxic substances such as carcinogenic, mutagenic or reprotoxic substances that can harm humans or the environment for short or long periods of time (eg, chemicals, radioactive substances)
- the sustainable development by recycling (eg, re-use of assemblies, raw material) or converting materials into energy (eg, burning with toxic gas processing)
- the demilitarization of defense Products to avoid weapons proliferation and use by terrorist groups, for example

This applies to:

- Product disposal at the end of the service life
- waste generated throughout the Product operation life cycle including, but not limited to:
  - ozone depleting substances (eg, Chlorofluorocarbon (CFC), Hydro chlorofluorocarbon (HCFC), halon)
  - global warming effect (sustainable development)
  - healthcare waste and sanitary waste
  - stores (eg, paints, solvents)
  - ballast water and sediments
  - anti-fouling paints (eg, biocides)
  - oily bilge water (sludge)
  - wastewater (black and grey water)
  - galley food waste
  - electronic scrap
  - complete equipment and equipment components
  - solid waste (eg, glass, paper, plastics)
  - exhaust emissions (eg, nitric oxide (NOx), sulfur oxide (SOx), carbon oxide (COx))
  - spares and consumables (eg, batteries, nuclear sources, oils)
- all the resources deployed and used to operate and support the Product (eg, infrastructure, facilities, support equipment)

Disposal activity for software must consider specific aspects. Software/data are always embedded in a piece of hardware. For that reason, the disposal of software requires appropriate methods to be successful and sustainable. Those methods include, but are not limited to:

- the removal of the software from the hardware (eg, tools to clean/clear memory). It is necessary to ensure that remaining bit structure within the hardware does not allow running or rebuilding the software by reverse engineering.

- the corruption of the software embedded in a piece of hardware, for example by physical destruction of junctions in a Field Programmable Gate Array (FPGA), or in a Read-Only Memory (ROM) module
- irrevocable destruction of the hardware carrying the software, by using sustaining hardware disposal methods like shredding, melting or etching

The disposal process influences the following project phases:

- In-service phase
  - necessary provisions to design safe disposal tasks against dangerous substances contained in the Product
  - disposal process of consumables and items removed as a result of operation and maintenance tasks (limiting their impact on health and environment)

  LSA data and the Product's technical publication must include this information as disposal tasks.

- Disposal phase
  - dismantling and disassembling
  - demilitarization
  - disposal of complete Product (eg, energetic recovery, material recovery, re-use, landfill waste)

  This information is available within the Product breakdown (information for parts) and Product Disposal File (PDF).

2 Disposal tasks during Product life cycle phases

2.1 Concept definition phase

A Product disposal strategy is created along with a concept of operation and a support policy. This strategy must make provisions to address the following matters appropriately:

- a sustainable design to limit the use of scarce resources, and the environmental impact of Product disposal
- the identification of applicable health and environmental regulations (refer to Para 5)
- a continuous survey (regulation watch) of list of black substances (refer to REACH regulation, appendix 17) and grey substances (refer to REACH regulation, appendix 14) that will be respectively forbidden and limited in use and quantity
- the method definition for recording and tracking dangerous substances that can be present in house and passed on to suppliers
- an estimate of Product disposal costs

2.2 Development phase

During development phase, the disposal activity is twofold:

- **Design the disposal:** This includes all plans made during the Product design with a sustainable development approach, such as limiting the use of scarce resources, and reducing the environmental impact of Product disposal (eg, no toxic substances, identification of materials for easy recycling). For this, a PDF is compiled (refer to Para 4). As part of Product data and configuration management, the Product breakdown records carcinogenic, mutagenic or reprotoxic substances used in the Product design. Chemical substances are identified by their Chemical Abstract Substance (CAS) code (or equivalent), according to the REACH European regulation requirements. As the Product design matures, the information in the PDF is updated.

- **Dispose of the design:** This includes all plans made during development to prepare further for the Product disposal phase including:

- perform a Product dismantling task analysis to identify and characterize the tasks required for Product disposal (eg, tasks duration, required resources). This can be considered as an extension of the Maintenance Task Analysis (MTA), using the same structured approach.
- assessing the impact of disposal on supportability elements, such as additional support equipment required to perform Product disposal, description of disposal tasks in technical publications, training courses to teach operators how to dispose of the Product (including health and safety precaution)
- estimate the cost of Product disposal. This can be a part of the Product global Life Cycle Cost (LCC) analysis, using the same structured approach.

2.3 In-service phase

During this phase:

– the PDF is updated to take into account:

- design changes in the Product during operation and support phases
- evolution of applicable regulations (eg, new banned substances, strengthening of health and safety regulations)

– resources required for Product disposal are produced and delivered
– disposal demonstration can be performed on Product prototypes to validate:

- the relevance and completeness of information compiled in the PDF
- dismantling and disposal operations (eg, people skills, technical publications contents, adequacy of support equipment)

Disposal of equipment to be replaced (eg, batteries, oils) according to the maintenance plan must occur in compliance with disposal requirements identified in the disposal analysis performed during the previous Product life cycle phases.

2.4 Disposal phase

During this phase:

– the disposal strategy defined during the concept definition phase is implemented to dispose of the Product (eg, dismantling, recycling, energetic valorization)
– the disposal operations are facilitated due to:

- a Product design that is disposal-oriented (design the disposal)
- an available set of resources developed to perform Product disposal (dispose of the design)

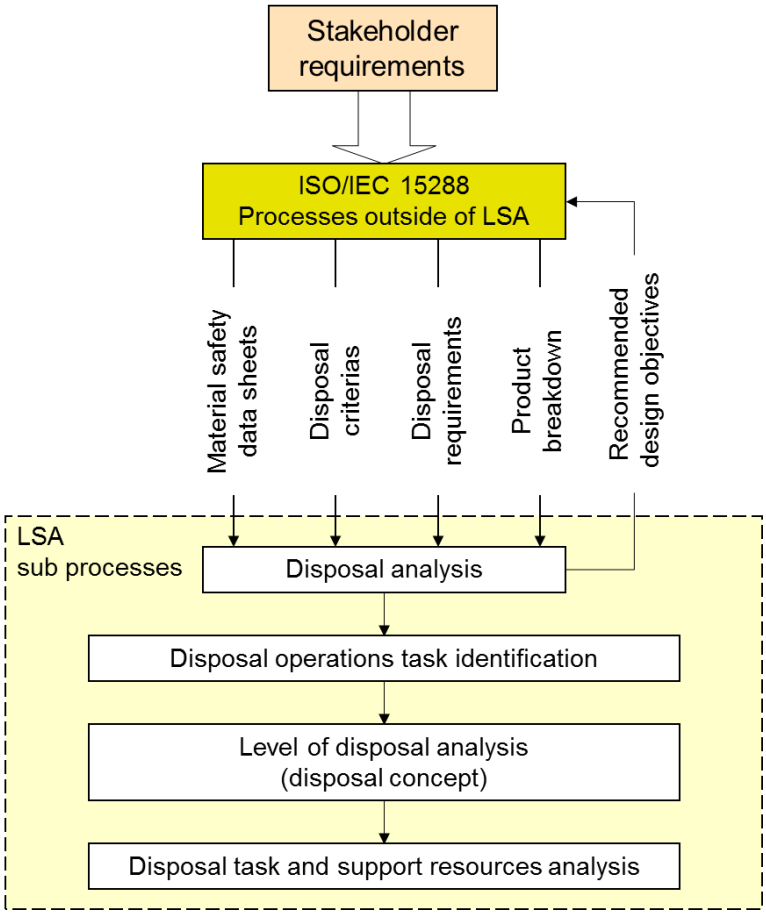
– collecting all lessons learned (eg, cost, best practice, things to avoid) to provide feedback for other projects

3 Disposal analysis

Disposal analysis needs to be tailored to the project needs (refer to Fig 1), and is the starting point of the LSA activities logic that determines the disposal process, disposal tasks, and support resources required.

This analysis applies to:

– Product components to be disposed of throughout the Product life cycle, as a result of the Product maintenance plan implementation
– the entire Product, at the end of its operational life

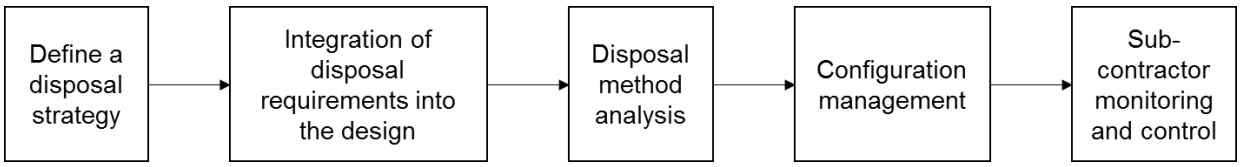


ICN-B6865-S3000L0089-002-01

Fig 1 LSA aspects of disposal analysis process

3.1 Disposal analysis activities

Fig 2 shows the disposal analysis logic that indicates which LSA activities to consider and tailor for the disposal at the end of a Product's life.



ICN-B6865-S3000L0090-001-01

Fig 2 Disposal analysis activities

3.1.1 Define a disposal strategy

Define the disposal criteria for each relevant item in the Product breakdown and establish a design solution.

3.1.2 Integration of disposal requirements in design

Upon the development of a new Product, the disposal requirements must be balanced against performance, reliability, safety and life cycle cost requirements.

Design principles must be a guide for design work and design considerations. To meet planned service life requirements, it is necessary to select all substances used in the design of the Product and its packaging.

3.1.2.1 Recommended materials
The selection of materials to be used in the design of the Product and its packaging must aim to minimize:

- hazards to personnel or property during preparation for disposal
- hazards to personnel, property or the environment as a result of disposal processes
- residual hazards from the materials which must be handled during the disposal process

It is mandatory to specify all materials and their associated hazards, in their primary state and during the stages of the disposal process. It is necessary to define material identification and marking processes, and include this information in the PDF (refer to Para 4).

3.1.2.2 Accessibility and extraction of components and hazardous materials
Provide design recommendations with requirements for a minimum of non-standard processes, tools and technical expertise for access and extraction, and identify any non-standard processes or tools needed to access components and hazardous materials, as well as any hazards associated with such processes.

3.1.2.3 Select components and materials to facilitate safe recovery and recycling
Identify the components and the base materials likely to be recovered from the Product and their recyclability. The assessment must consider possible degradation and contamination of components and materials during their service life to ensure that recovery and/or recycling is viable.

3.1.2.4 Complete material list
List all materials used in the Product in a supplier material list and make it available to all personnel involved in the project. The list must include, but is not limited to:

- components
- materials
- compatibility
- residual products used by the Product
- residual products at planned disposal
- impact on the environment at planned disposal
- applicable health and environmental regulations

3.1.3 Disposal method analysis
Disposal method analysis includes:

- the identification of potential alternative applications (re-use) for the Product and its components, including software. The analysis demonstrates that such alternative applications are technically achievable, environmentally acceptable, and economically viable.
- the identification of components and material that must be recycled
- the identification of components and material that must be recovered
- the identification of components and material that will be land waste
- for each alternative above, definition of the residual products. Refer to Table 2.
- a report on the results concerning disposal attributes in the Product breakdown within the part information
- disposal of sensitive information: If Products to be disposed contain proprietary, confidential or classified software and data, the dismantling procedure includes:

- the transfer of the necessary historical information to a media or other storage device for proper preservation or historical archiving
- the destruction of the data/software media or the erasure of the sensitive, proprietary, and classified information and software in an adequate manner so as to ensure that such information/software cannot be recovered by third parties, in accordance with the proper security regulations of the customer / the Product owner

Table 2 Residual product

Residual products	Re-useable/ recyclable [g]	Combustible [g]	Waste [g]	Accumulated weight [g]
Sum	20	2000	3000	5020

3.1.4 Hazards

Identify all hazards associated with the design and in particular:

- providing an estimate of the likely levels of noise, flash, and toxic or corrosive fumes produced at the time of operation, firing or launching
- identify any disposal hazards associated with the Product design that require additional safety arrangements other than the existing disposal resources
- identify any residual hazards from the disposal of the Product that require additional safety arrangement other than the existing disposal resources
- considering possible means of fail safe
- identify any features that can pose an unexpected hazard to operatives taking care of explosive ordnance disposal. These features include, for example, anti-tampering/interference fuses or energetic materials with particularly high levels of sensitivity. This assessment must consider the effects of aging on the Product.

3.1.5 Configuration management

For any component/design change, assess its impact on the:

- Product life extension
- reduction of health and environment hazards

Demonstrate that such changes are both technically achievable and economically viable.

3.1.6 Monitor and control of sub-suppliers

Product elements obtained from suppliers must meet disposal requirements. This effort applies to equipment obtained from any supplier, regardless of tier. Review all sub-supplier contracts and perform evaluations of their disposal procedures.

The same requirements for ordinary suppliers apply also in the case of GFE, which is comparable to ordinary sub-supplier item.

During the subsequent phases of the development program, sub-supplier specifications must incorporate aspects such as:

- disposal requirements
- design constraints and requirements
- Safety Data Sheet (SDS) for Europe, or Material Safety Data Sheets (MSDS) for USA

3.1.7 Task input

The inputs for the identification of the disposal analysis activities are the:

- disposal strategy
- disposal requirements

- disposal criteria
- Product breakdown
- SDS or MSDS detailing:
  - the Product name used on the label, chemical and common names of ingredients that are known to be health hazards and constitute 1% or more of the composition. Carcinogens must be listed if their concentration equals or exceeds 0,1%.
  - the chemical and common names of all ingredients that present a physical hazard when present in the mixture
  - relevant physical and chemical characteristics of the hazardous chemical (eg, vapor pressure, flash point)
  - relevant physical hazards, including potential fire, explosion, and reactivity hazards
  - relevant health hazards, including signs and symptoms of exposure, and any medical conditions generally recognized as being aggravated by exposure to the chemical
  - the primary way of entry into the body
  - the Occupational Safety and Health Administration permissible exposure limit and American Conference of Industrial Hygienists threshold limit value. It is possible to list additional applicable exposure limits.
  - whether the hazardous chemical is listed in the National Toxicology Program Annual Report on Carcinogens (latest edition) or is considered a potential carcinogen by the International Agency for Research on Cancer Monographs (latest editions), or by Occupational Safety and Health Administration
  - precautions for safe handling and use, including appropriate hygienic practices, protective measures during repair and maintenance of contaminated equipment, and procedures for clean-up of spills and leaks
  - appropriate control measures, such as engineering controls, work practices, or personal protective equipment
  - emergency and first aid procedures
  - the date of preparation of the MSDS or the last change to it
  - the name, address and telephone number of the chemical manufacturer, importer, employer or any other responsible party that prepares or distributes the MSDS, and that can provide additional information on the hazardous chemical and appropriate emergency procedures, if necessary

3.1.8 Task output

The outputs for the identification of the disposal analysis activities are:

- scoping the disposal business process to be applied
- listing a "disposal Product breakdown", detailing which items are suitable for re-use, recycling, recovery or landfill waste

3.2 Disposal operation task identification

3.2.1 Disposal process breakdown analysis

Define a representative set of activity sequences to identify all required services that correspond to anticipated disposal process of a Product and environments.

The task must list the method recommended for disposal of the Product items. Disposal involves two aspects:

- dismantling the Product
- disposing of materials and contaminated items, including contaminated solvents

As details of the design become available, it is mandatory to update the breakdown of the recommended process.

This requires the use of the following data elements:

- disposal task identifier
- disposal recovery codes for the Source Maintenance & Recoverability (SMR) code, coding for the disposal method for the Product (eg, recycling, energy recovery, landfill waste)

3.2.2 Task identification

Identify and document the activities/tasks for Product/component disposal. The level of identification of these tasks must be commensurate with design, including disassembly diagrams, step-by-step procedures, safety precautions and component and material tables, taking into account possible changes due to aging, and final destination of all Product components. It is mandatory to identify any special tool and equipment required and any limitation on locations.

A task inventory is necessary for the Product and its components. It is recommended to include this task inventory in the LSA data. Based on the disposal process breakdown analysis, and scenarios/conditions, the task inventory must identify all tasks that operators, maintainers, or support personnel must perform with regard to disposal of the equipment/components (hardware and software). The level of detail for tasks must be commensurate with design and the disposal scenario development. The task inventory must include a task classification that defines scenario/conditions, function, job and duty, task and subtasks and task resources. The task inventory must include task descriptions, each of which consists of:

- an action verb which identifies what is to be accomplished in the task
- an object which identifies the task object

Task descriptions must be clear and concise. Moreover, they must identify hazardous materials, generation of waste, release of air and water pollutants and environmental impacts associated with each task. In case the same task pertains to more than one job, and therefore identified in a collective task for training purposes, the task will be identified as a collective task within the task inventory. Task classification must define all verbs unambiguously.

3.2.3 Disposal hazard analysis

The hazard analysis of the chosen disposal process must identify the hazards related to the process and its products, and must classify the associated risks, eg, by defining risk scores and risk weighing. Examples are given in Table 3.

- 2 = Serious
- 1 = Moderate
- 0 = Harmless

Table 3 Risk score examples

Risk score	**Type of risk**	**Weight**	**Hazards from planned disposal process**	**Total** (Risk x Weight)
2	Safety	3	Detonation of warhead during dismantling of missile	6
1	Health	3	Toxic gases produced by the combustion of launch rocket engine propellant	3
1	Environmental	2	Environmentally polluting gases produced by the combustion of launch rocket engine propellant.	2

The hazard analyses of the Product must include a description of the Product with its configuration. The analyses must list the composition and mass of each material, along with an aggregate mass of all materials in the item. It is necessary to identify the hazards associated

with each material, along with any special handling requirements. Analyses must describe both primary hazards and secondary hazards (those derived from material changes that occur as a result of the processes used in disposal). All materials to be transported must be accompanied by safety data sheets.

Risks to be listed and considered are, for example:

- **Explosive hazards:** Risks associated with explosive materials in the Product
- **Fire hazards:** All flammable and oxidizing materials
- **Caustic hazards:** All caustic materials
- **Ingestion, absorption, inhalation and adsorption:** All materials that have a physiological effect on personnel or may damage property. The list must indicate the physical form of the material, route of entry and the required protection.
- **Trauma inducing:** All non-explosive energy stores such as batteries, electrical and electromagnetic devices, capacitors, sources of static charge, springs under tension or compression, and any material that, when released, can transfer an amount of energy sufficient to cause trauma. The list must indicate the expected energy output and effect, and the required protection.
- **Environmental hazard:** Any potential environmental hazard not already covered, including potential damage to the atmosphere, soil and groundwater.

3.2.4 Activity input

The inputs for a task identification analysis for disposal are:

- the delivery identification of any data item required (eg, to be documented in the LSA Guidance Document (LSA GD))
- the identification of equipment hardware, software and firmware on which the task will be performed, and the level of indenture up to which this analysis is performed
- the identification of the levels for disposal of hardware, software and firmware analyzed during this activity, with the aim to identify operations and tasks
- documentation requirements over and above the LSA data, such as functional flow diagrams or drawings
- a description of Product concepts under consideration
- the disposal methods

3.2.5 Activity output

The outputs for the disposal task identification are:

- a task inventory documented in the LSA data, or equivalent format approved by the project, identifying disposal task requirements. Task inventories must include task descriptions on Product hardware, software and firmware, and must indicate the levels of indenture specified by the project
- the identification of any hazard risks involved in satisfying the requirements for disposal operational tasks

3.3 Level of disposal analysis

Hazardous substances location in the Product breakdown will determine the level of disposal. The Product must be dismantled and disassembled to a level that allows the use of the same disposal method as defined in Para 4.

3.4 Disposal task analysis

3.4.1 Task description

The detailed disposal task analysis based on the principles of MTA includes the following activities:

- conducting a detailed analysis of each operation, maintenance and support task contained in the task inventory and determine:
  - task resources required to perform the task (considering all IPS elements)
  - elapsed time and man-hours for intended disposal facility/environment for the Product
  - maintenance level assignment based on the established support plan
  - environmental impact of the tasks, including use of hazardous materials, generation of hazardous waste and its disposal, and the release of air and water pollutants
- identify new or critical task resources required to perform each task, and the hazardous materials, hazardous waste and environmental impact requirements associated with those resources. New resources require development in order to disassemble the new Product. These resources can include support and test equipment, facilities, new or special transportation products, new computer resources, and new test or inspection techniques. Critical resources are not new, but the management must pay special attention due to schedule constraints, cost implications, or known scarcities. Unless otherwise required, it is necessary to document new and modified support resources in the LSA data, or with equivalent documentation approved by the project, in order to provide a description and justification for the resource requirement.
- conduct a transportability analysis on the Product and any section thereof when disassembling is required for transport
- document the results in the LSA data, or in an equivalent format approved by the project

3.4.2 Task input

The inputs for a disposal task analysis are:

- identification of Product hardware and software on which the analysis will be performed
- identification of indenture levels up to which the analysis will be performed
- identification of the levels of disposal operations documented during performance of this task
- known or projected shortages of task resources
- any requirement for supplemental documentation over and above LSA data (eg, transportability clearance diagrams)
- delivery identification of any required data item
- any information available from the project relative to:
  - existing and planned personnel competences, capabilities, and training program
  - lists of standard support and test equipment
  - available facilities and infrastructure
  - available training devices
  - existing transportation products and capabilities
- description of personnel capabilities (target audience) intended for Product disposal at each level of disposal
- operations and disposal task requirements from the task (disposal operations task identification)
- recommended disposal plan for the Product from the task (level of disposal analysis)
- supportability and supportability-related design goals and requirements

3.4.3 Task output

The outputs of the disposal task analysis are:

- completed LSA data or equivalent format approved by the project on Product hardware and software to the level of indenture specified by the project
- identification of new or critical support resources required to operate and maintain the new Product

- alternative design approaches in case tasks fail to meet established goals and constraints for the new Product, or there is an opportunity to reduce disposal costs or optimize support resource requirements
- identification of management actions to minimize the risks associated with each new or critical support resource requirement
- validation of key information documented in the LSA data
- output summaries and reports as specified by the project, containing all pertinent LSA data at the time of preparation
- LSA data that is updated as soon as better information becomes available and as applicable input data from other Product engineering programs are updated

4 Product disposal file

The purpose of PDF is to compile all data information regarding Product disposal. It is mandatory to update the PDF throughout the Product life cycle, in order to reflect Product changes (configuration management), evolution of maintenance plans, and health and environmental regulations modifications.

This file must allow the Product project management to clarify the operational limits of the authorized substances over a given period, in order to anticipate the research of alternative solutions that are more suitable in terms of:

- sustainable development
- health and safety impact on humans and environment

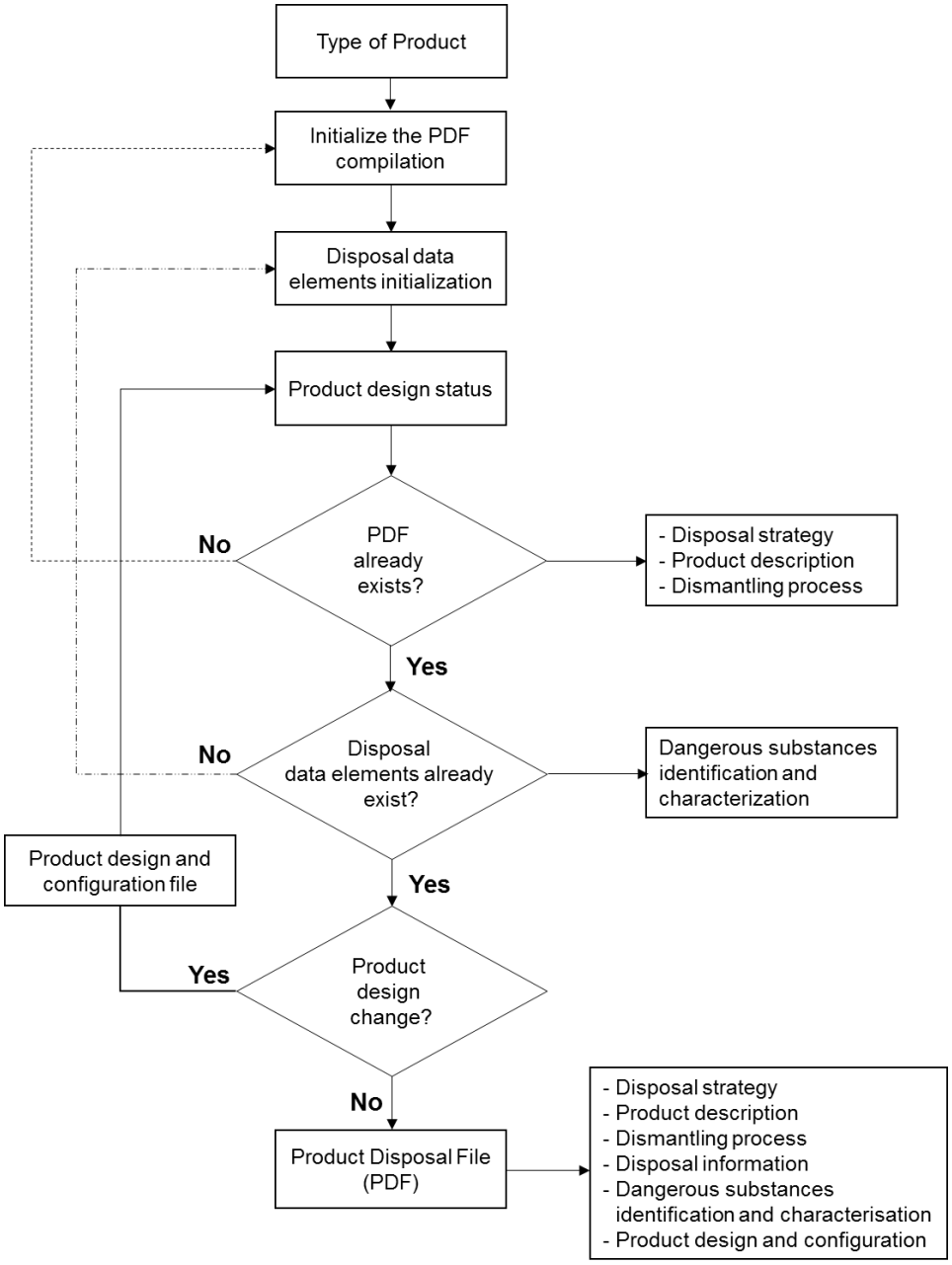
4.1 Product disposal file content

Whatever the Product, the PDF can compile information including, but not limited to:

- Product breakdown and associated functional and physical description
- identification of components, products or ingredients (eg, containing hydrocarbon, lubricating, paintings) related to Product breakdown (with illustrations wherever possible)
- a list of risks for people and the environment for each dangerous substance identified in the Product
- identification of the possible processing for each type of Product component at disposal, according to the type of technology (eg, electronic, mechanic, hydraulic and pyrotechnic) and the contained substances:
  - components immediately reusable on another Product
  - components containing high value material (eg, gems, precious metals)
- methods of elimination that will differentiate potentially harmful substances and harmless material
- conditions for disassembly and the destination of the components (eg, re-use, recycling, energetic valorization)
- disassembly procedures down to the required level of depth, to allow all materials (harmful or hazardous materials in particular) to benefit from the processing and their method of elimination
- rules, instructions, measurements or provisions that guarantee a sufficient level of safe practices
- collection of all safety data for all the harmful or hazardous materials composing the entire Product
- detailed, step-by-step procedures for dismantling and disposal
- a list of proposed organizations certified for proper recycling and elimination of the dangerous residues, according to the applicable regulation

4.2 Product disposal file compilation process

The flowchart in Fig 3 shows the PDF compilation process.



ICN-B6865-S3000L0091-002-01

Fig 3 PDF compilation process

4.3 Product disposal file data elements

This paragraph presents a proposed list of data elements to be compiled in the PDF. This list is not restrictive and must be agreed to on a case-by-case basis (eg, within LSA GC), depending on the context of each specific project and contract.

In the first column of Table 4, text in *italics* refers to potential data element descriptions within a PDF (refer to Fig 4).

In the second column of Table 4, text in *italics* refer to S3000L data elements (if applicable) which can potentially cover the data requirement from a PDF (refer to Chap 22).

Table 4 Proposed PDF data elements

Header in PDF Meaning in PDF	Meaning *corresponding S3000L data element, if available*
Subsystem	Identifies the sub-unit to which the item belongs in the Product.
Assembly identifier	*breakdownElementIdentifier*
Item	Name of component according to drawing
Part name	*partName*
Article number	Refers to drawing identifier for unequivocal identification or other article number (for some components such as screws and adhesives).
Part number	*partIdentifier*
Material denomination	Short description of material in component (if known)
Material description	*substanceDescription*
CAT	The main substances category
Material category	*substanceUsageCategory* Examples: – black (forbidden) – grey (authorized with limitation) – green (authorized with no limitation)
CAS	CAS code
Material code	*substanceIdentifier*
Phrase of risk	Phrases of risk (in accordance with REACH regulation (EC) No 1907/2006 on Safety Data Sheet data)
Type of risk	*substanceRiskDescription*
MSDS ref.	Reference of the associated Material Safety Data Sheet Document classified as "Material Safety Data Sheet", assigned to the Material Class as "Reference"
MSDS	Linked as a referred document using the data structures defined in the Unit of Functionality (UoF) Document, refer to Chap 19.
Service life	Product service life
Service life	*hardwarePartOperationalAuthorizedLife*
Mass [g]	Weight of component (in grams)
Mass	*quantityOfContainedSubstance*
Reg.	Applicable environmental regulations

Header in PDF Meaning in PDF	Meaning *corresponding S3000L data element, if available*
Regulations	Linked as a referred document using the data structures defined in the UoF Document, refer to Chap 19.
SLA No. Serial number	Serial number of components not applicable
Risk factor (SLA) Risk criticality	Judged risk factor (criticality) of component (scale 1 to 4) *hardwarePartHazardousClass* in relation with *substanceRiskFactor*
Critical property Justification	The most important property for the function of the component *containedSubstanceJustification*
Proceedings in connection with use Disposal concept in use	Describes component disposal during the in-service phase. Documented in the UoF which cover all the data of a maintenance task, refer to Chap 19.
Proceedings in connection with planned disposal Disposal concept end of life	Describes component disposal during planned disposal. Documented in the UoF which cover all the data of a maintenance task, refer to Chap 19.
Waste products in connection with use Waste products in use	Indicates the waste products from component disposal according to the procedure during use or after use. *hardwarePartWasteProductsInUseDisposalDescription*
Waste products in connection with planned disposal *Waste products disposal*	Indicates the waste products from component disposal according to the procedure for planned disposal. *hardwarePartWasteProductsPlannedDisposalDescription*
Environmental aspects, use Environmental aspects in use	Classification of environment aspects during use. Refer to Table 5 for the "environment aspect codes". *hardwarePartEnvironmentalAspectInUseClass*
Environmental aspects, planned disposal Environmental aspects end of life	Classification of environment aspects at end of life. Refer to Table 5 for "environment aspect codes". *hardwarePartEnvironmentalAspectPlannedDisposalClass*
Task	Dismantling tasks characterization and description, identifying required resources and associated safety cautions to preserve human beings and the environment.

Header in PDF Meaning in PDF	Meaning *corresponding S3000L data element, if available*
Task requirement	Documented in the UoF which cover all the data of a maintenance task, refer to Chap 19.
Last updated SLA/environs	Date of latest changes to the environmental part of the SLA
Date	not applicable

Table 5 Environment aspect codes

Code	**Meaning**
tox	Harmful to the environment (toxic, bioaccumulation and/or difficult to decompose)
acid	Acidification
ozon	Dangerous for the ozone layer
gwp	Greenhouse effect
mw	Material waste
pos	Environmental aspects, positive
ene	Energy regaining by burning
mr	Material recycling

Item	Article number	Material denomination (specification)	CAT	CAS	PHR risk	MSDS ref.	Service life	Mass [g]	REG	SLA No.	SLA risk	Critical Property (important for service life qualification)	Proceedings in connection with		Waste products in connection with		Environmental aspects		Task	Last up-dated SLA/ Environ
													Use	Planned Disposal	Use	Planned Disposal	Use	Planned Disposal		
Outside tube	4113499	PC 10% glass fibre Black						58,0		1	4	Mechanical strength	Thrown	Burned/ recycling	Stable	COx /PC	mw	ene, gwp/mr		04/03/27
Plug	5229068	PC 10% glass fibre Black						1,1		2	4	Mechanical strength	Thrown	Burned/ recycling	Stable	COx /PC	mw	ene, gwp/mr		04/03/27
Support Stand	4113498	PC 10% glass fibre Black						26,7		4	4	Mechanical strength	Thrown	Burned/ recycling	Stable	COx /PC	mw	ene, gwp/mr		04/03/27
Slotted screw MCS M4x25	6436317	Stainless Steel ISO 1207						3,0		10	1	Mechanical strength	Thrown	Recycling	Steel	Steel	mw	mr		03/01/30
Connector cover	2799541	Aluminium EN 1706 AC42000-T6 Chromated Cr						5,0		368	2	Mechanical strength	Thrown	Recycling	Al	Al	mw	mr		03/01/30
Adhesive	10363883	Loctite 406						0,0		3	4	Adhesion	Thrown	Burned	Stable	COx, NOx	mw	ene, gwp, acid, ozon		03/01/30

ICN-B6865-S3000L0092-003-01

Fig 4 Example of a PDF (for a given subsystem of a Product)

5 List of regulations

This paragraph identifies international regulations for health, safety and environmental conditions. It is necessary to define the applicability status of the regulations on a case-by-case basis for each specific project and contract.

5.1 European Union directives

- European Community Directive (2012/19/UE), Waste of Electrical and Electronic Equipment (WEEE), in force since 2012, can be used as a guideline for target values for recovery, recycling and re-use
- European Community Directive (2006/121/EC), Registration, Evaluation and Authorization of Chemicals (REACH), concerning the recording, evaluation and authorization of chemical substances, as well as the restrictions applicable to these substances, and establishing a European Chemicals Agency
- European Community Directive (2002/95/EC) of the European Parliament and the Council of 27 January 2003 on the restriction of the use of certain hazardous substances in electrical and electronic equipment

5.2 Transportation regulations

National/state-run competent authorities must approve transportation of products recommended to be returned for safe disposal. The authorities will issue transport certificates stating the component's transportation classification based on the UN Recommendation on the Transport of Dangerous Goods (known as the "orange book"). This international recommendation classifies dangerous goods based on the type of risk, UN classification code, product category codes, and UN number.

The UN number defines the packing group and packing instruction for the Product, maximum weight of the package, and how to label the package based on the mode of conveyance. Note that the packing itself needs to have type approval issued by a competent authority.

The recommendations need to be tailored for transportation by road, railway, sea and air, including:

- an agreement on Dangerous Goods by Road (Europe)
- regulations concerning the International Carriage of Dangerous Goods by Rail (European law)
- IATA Dangerous Goods Regulations - international recommendation for air transports
- International Maritime Dangerous Goods Code (United Nations) for sea transports

Transport certificates are often valid at a national level and they are normally valid for a maximum of 10 years. At the time for disposal, new certificates must be applied for at the competent authorities concerned.

5.3 Ground vehicles

Directive 2000/53/EC of the European Parliament and of the Council on end-of-life vehicles serves as the base for the PDF.

This directive is valid only for civilian vehicles weighing 3.5 tons or less. This directive is the basis for the development of a future regulation that will include all civilian and military vehicles. This directive envisages certain measures and incentives so that the dismantling phase is taken into account from the beginning of the Product design. In particular, it will be based upon new standards deriving from scientific progress. The principal objective of this directive is to protect humans, animals and the environment through a better management of vehicle use. A disposal file ensures improved vehicle management by providing adequate solutions for the assumption of the waste responsibility or the use of replacement materials that are less harmful to the environment.

5.4 Navy equipment

The PDF must comply with the requirements of the "green passport". These principles are stated in IMO Resolution No A.962(23), dated December 5, 2003, as amended by Resolution No A.980(24) and Resolution A981(24), dated December 1, 2005. The latter is included in international texts currently in force, such as the *Basel Convention on transboundary movements of Hazardous Wastes and Their Disposal*.

The A.962(23) resolution is applicable to any maritime equipment including components with a "green passport". The green passport is an instrument that aims to facilitate dismantling and recycling operations for any ship that arrives at the end of its lifetime. This is possible by adopting good practices for the entire life cycle of the ship (respectful of environment on the level of the dismantling site and making safe for the personnel carrying out this work), with an aim at reducing progressively the use of any dangerous materials throughout the ship service life.

In order to have further information on the "green passport", refer to document NR528 published by Bureau Veritas.

The directives are intended to provide the flag states, port states and re-user states, owners of ships, manufacturers of ships, suppliers of navy material and recycling installations with several indications on the "best practices", which take into account the ship recycling process throughout the ship life cycle.

They take into account certain codes and protocols relating to environmental protection.

The list of the substances to be tracked is based on Directive 67/548/CEE of the European Parliament and the Council, and the States can suggest adding other substances for the drafting of the convention.

5.5 Air equipment

Currently, there are no regulations or guidelines accepted worldwide for aircraft disposal at the end of their service life.

5.6 Ammunition

The PDF will comply with the STANAG 4518 edition 1 (2001), Safe Disposal of Munitions, Design Principles and Requirements, and Safety Assessment.

The goal of this NATO standardization agreement is to provide the NATO members with a shared source of principles and requirements of safety for the design and the evaluation procedures, to ensure safe disposal of munitions.

5.7 Nuclear waste

Each member state of the International Atomic Energy Agency (IAEA) has its own standards to manage nuclear waste.

However, the IAEA released the following documents (Safety Guides and Information Circular) addressing nuclear wastes safety:

- Environmental and Source Monitoring for Purposes of Radiation Protection (RS-G-1.8)
- Management of Waste from the Use of Radioactive Materials in Medicine, Industry, Agriculture, Research and Education (WS-G-2.7)
- Code of Practice on the International Transboundary Movement of Radioactive Waste (INFCIRC/386)

Moreover, the Board of Governors approved a safety requirements publication on geological disposal (SSR-5), co-sponsored by the Organization for Economic Co-operation and Development/Nuclear Energy Agency (OECD/NEA).

For further information, refer to Table 6, which includes a list of international websites.

Table 6 Websites for additional information concerning disposal of nuclear waste

Site address	Organization
http://www.nea.fr/	Nuclear Energy Agency with 28 member states
http://www.iaea.org/	International Agency Energy Atomic with 144 member states
https://www.unscear.org/	United Nations Scientific Committee on the effects of atomic radiation

6 Associated parts of the S3000L data model

The documentation of the associated data to this chapter is supported by the following Units of Functionality (UoF), refer to Chap 19:

- S3000L UoF LSA Candidate
- S3000L UoF Part Definition
- S3000L UoF Task Requirement
- S3000L UoF Task Usage
- S3000L UoF Time Limit

Chapter 17

In-service LSA

Table of contents

List of tables

List of figures

References

Table 1 References

Chap No./Document No.	Title
Chap 12	Maintenance task analysis
Chap 19	Data model
S4000P	International specification for developing and continuously improving preventive maintenance
S5000F	International specification for operational and maintenance data feedback

1 General

Any support solution developed and implemented using the LSA methodology during the early phases of the Product life cycle and during the design and development phase, usually meets customer requirements concerning Product availability and Logistic Support Costs (LSC). The solution must be effective and sustainable. However, especially for technically complex Products with a long operating life, many different aspects will influence the support solution implemented during the in-service phase of a Product. The support solution must consider all these aspects to guarantee a proper support during the complete life cycle of the operated Product.

In-service LSA takes into consideration the:

- handling of design changes (engineering changes) of the Product itself, for example changes driven by Product improvements implemented by industry, due to obsolescence or additional/modified technical functionality requested by the customer
- handling of changes of the support scenario (eg, driven by a role change for the Product or a deployment to a different climatic environment) or of the operator's support capabilities (eg, reduced availability or even closure of maintenance facilities, changes in the customer organization, reduction of staff)

- identification of required optimization of the support solution following the operator's feedback (eg, significant increase of LSC, supportability KPI are not met, problems during the performance of support tasks)

The in-service LSA activities are part of the Product life cycle activities. After the Product enters into service, it is recommended to collect, document, and prepare for a later evaluation all the in-service data that are relevant to evaluate the efficiency of the support solution. Those data will be subject to an analysis process against defined support requirements. Those support requirements can change several times during Product life cycle, and problems or shortcomings appearing at later stages can influence such requirements. The LSA methodology for the in-service phase must be adapted accordingly, to ensure a continuous improvement of the support solution for any Product, and taking into consideration all potential changes and the operator's feedback.

As it is possible to conduct several in-service LSA iterations throughout the in-service phase, it is vital to put in place a Configuration Management (CM) process for the results of the LSA activities. This includes a thorough configuration control, accounting and audit. In addition, the in-service LSA relies on an operation and maintenance feedback system that needs to be in place when the Product enters into service.

1.1 Purpose

This chapter aims to provide a methodology for defining the necessary in-service LSA activities. This includes a process to monitor, evaluate and adapt a support solution during the in-service period of the Product life cycle. This process is described as the In-Service Support Optimization (ISSO) process.

Prior to a Product’s entry into service, it is necessary to define all support tasks and the resources required to support operation and maintain the Product. Reviews of the support solution help monitoring the technical applicability and the cost-effectiveness of the current support solution. In case of inadequacy in technical or economic terms based on suitable in-service data, it is necessary to identify, recommend and implement potential modifications to the support solution, if applicable. All these activities contribute to ensure that:

- the Product and its support solution are properly maintained and can therefore satisfy any support contracts
- the Product availability can be sustained or enhanced
- the logistic footprint and the LSC can be kept within reasonable limits
- the Product availability and LSC are reasonably balanced
- the support solution is periodically reviewed and optimized, taking into account any changes to capability, operational or environmental requirements
- the support solution continues to meet the operator's capability requirements
- the support solution is routinely reviewed to ensure it meets all applicable legislative requirements

Implementing LSA activities during the in-service phase is a win-win approach for both customer and contractor, because:

- the contractors will have a deeper knowledge on the actual Product use, and will be able to anticipate Product support issues, therefore improving their ability to solve support issues effectively
- the operator will benefit from the continuous improvement brought by the contractor on the Product supportability and on the support system efficiency

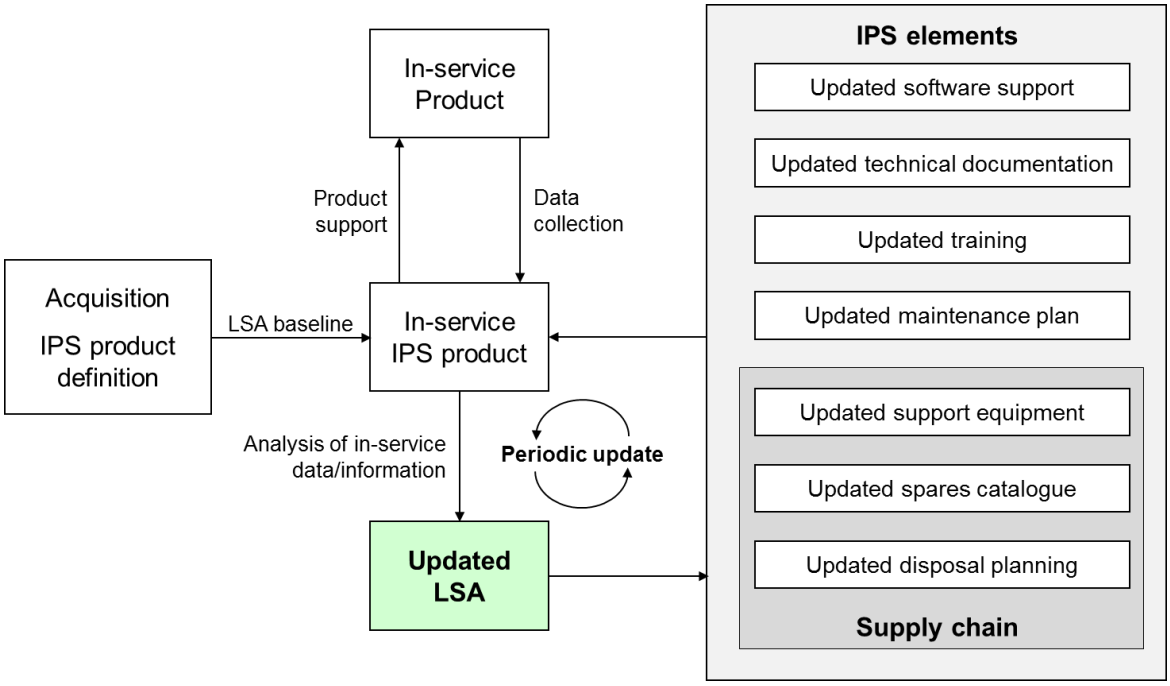
1.2 Scope

This chapter is applicable to the Products that enter into service and for which LSA has been performed prior to their entry into service. It is necessary to determine on a case-by-case basis the in-service LSA activities for the Products for which LSA has not been performed in accordance with S3000L. The depth and extent of the in-service LSA activities will depend on

the depth and quality of in-service data/information available and on the scale of changes. The drivers of these changes can be:

- Product modification and mid-life updates, sometimes referred to as "technology insertion". This option predetermines specific points during the Product life that, when reached, will prompt an update of the Product design or of some Product parts, as well as an update of the corresponding support solution. Replacement of obsolete items will also occur.
- changes to operational or environmental requirements, or changes to the operational and environmental conditions. It is necessary to assess those changes to identify their impact on the current support solution. The analysis can result in changes to the Product or to the support solution.
- operator's feedback on the effectivity of the current support solution (eg, problem reports, relevant support data to evaluate the as-is against the as-predicted situation)
- a need to review changes in the legislation. It is necessary to assess any impact on the current support solution.

Taking into consideration all the aspects listed above, the consequence can be a potential modification within the support solution. It is necessary to document such modifications. LSA data provide the baseline information concerning the support solution. The first step is to analyze any potential changes. In case the changes lead to a modified support solution, it is necessary to implement the changes into the LSA data. Consequently, the IPS elements forming the support solution require an update according to the changes within the LSA data. Following this approach, it is possible to trace the IPS elements throughout the Product life cycle. Fig 1 shows the top-level in-service IPS process.



ICN-B6865-S3000L0109-001-01

Fig 1 In-service LSA process

2 Core principles of in-service LSA performance

2.1 Product modification requirements

The main aspects already mentioned in Para 1.1 have different characteristics. The first aspect is to implement modifications to the support solution, based on engineering changes to the Product itself. Usually, supportability aspects do not initiate those changes. The requirements that can initiate those changes include, but are not limited to:

- a decision to implement new equipment into the Product, due to:
  - requirements coming from the customer for new or improved functionality of the Product
  - need for equipment or equipment components replacement due to obsolescence or due to changed legislative requirements (eg, the use of specific substances is prohibited)
  - improvement driven by industry due to technological progress
- a decision to modify/improve the Product in any other way, for example:
  - modification of software for equipment within the Product
  - modified installation location of equipment within the Product
  - decommission of equipment within a Product
  - new or improved surface protection
  - modification of structural components (eg, strengthening, modified material)

The implementation of modifications such as those listed above occurs after customer and contractor agree on implementing the changes. Typically, those modifications occur during Product retrofit or overhaul activities. Product optimization by design changes of hardware or software can require some adjustments to the support solution. The LSA process accompanies the initial design and development process, and it indicates how to identify a proper support solution for the modified Product. New or modified equipment can lead to new or modified support tasks and to relevant changes within the IPS elements.

It is necessary to harmonize the effort to update LSA data and the corresponding IPS elements during the in-service phase with the design process, which considers the existing support solution when developing a new design solution. The process follows the regular IPS process, except for potential design constraints coming from the existing in-service support solution.

The LSA data must document the consequences that Product engineering changes have on the support solution. This includes documenting the changes to the Product breakdown, new or modified task requirements, and new or modified support tasks following changes in the design and the relevant MTA. The configuration management of LSA data must be a basic principle to keep under control the different potential variants/revisions of breakdown elements (including their realizations by different parts), task requirements, maintenance tasks and operational support tasks.

All the changes in the LSA data must be incorporated to the IPS elements accordingly.

Note

Apart from the consequences of Product modification on corrective maintenance and operational support tasks, and on the performance of preventive maintenance tasks, it is necessary to evaluate whether there is a general impact on the preventive maintenance program for the Product. Are there any additional PMTR? Is there a need to modify the existing PMTR? Refer to S4000P, which covers initial PMA and the ISMO process.

2.2 In Service Support Optimization (ISSO)

The second aspect of the core principles is handling issues requiring a deeper analysis based on the operator's experience and capability in the context of Product maintenance and operational support. These issues can lead to reduced availability, to extended effort for maintenance or operational support and, in this context, to increasing costs and customer dissatisfaction.

Whenever the customer realizes and accepts an ongoing evaluation process of the actual support solution, this triggers the performance of an ISSO process. It is possible to perform the analysis after specific time intervals of Product use. The customer can contact the industry to analyze the actual maintenance and operational support concept and to provide recommendation on how to update the support solution for a better performance.

The currently implemented support solution provides the base for any subsequent issues. This support solution needs a task-by-task evaluation concerning its effectivity and applicability. The evaluation process is called the ISSO process. It offers a structured approach to optimize the support solution of a Product after the design and development phase, and it aims to obtain the required balance between costs and availability of the Product during a specified period of time or use phase. It is possible to use the process for every phase of the Product's life, after the creation of the initial support solution. This includes the possibility to cover the end of the Product's life or even during its disposal. A run-down plan to put a Product out of service can change policies. Some systems must be kept in running condition while the entire Product is being dismantled.

The ISSO process is different from the ISMO process, which is described in S4000P. The ISMO process deals with the preventive maintenance activities. In this context, the evaluation of PMTR determines whether the PMTR is still valid or can/must be modified. Refer to S4000P.

2.2.1 Optimization requirements following operator's decisions

Beside the modification of the Product itself, changes initiated by the customer, and/or the operator, can influence the support solution. Examples of decisions that can lead to a modification of the support solution include, but are not limited to:

- role/functionality change for the Product in use
- change in use environment, including deployment and change in geographical location or distribution of Products
- changes to task resources (eg, use of new support equipment or consumables, change of staff structure due to obsolescence of competences, closure of maintenance facilities)
- strategic decision of the customer/operator to change the maintenance strategy (eg, no repair activities for specific Product components on customer site any longer ⇨ selected repair tasks shifted to industry)
- strategic decision of the customer/operator to modify other parts of the in-service support solution with respect to operational support (eg, a shipping company is in charge of specific transport tasks for Product components, and the operator no longer performs them)
- modification of the maintenance concept by including new technologies (eg, including or removing sensors for condition-based maintenance)

Typically, the customer/operator initiates those modifications and industry/supplier does not influence this type of decision. The consequence is often a modification of the support solution. This means that support tasks change and require an update and/or integration in the LSA data. The configuration management of LSA data must be a basic principle to control the different potential variants/revisions of breakdown elements (including their realizations by different parts), task requirements, maintenance tasks and operational support tasks. It is necessary to incorporate all the changes in the LSA data into the IPS elements accordingly.

2.2.2 Optimization requirements following the analysis of actual support solutions

In general, the second goal of an ISSO process is to identify a potential for optimization and to resolve any shortfalls in Product maintenance and operational tasks performance. Below are some examples that show the need for optimization:

- Increasing costs:
  It is necessary to determine which costs can be attributed reasonably to Product support, and to identify cost drivers. Request data/information from operators and/or maintainers.

- Insufficient availability of the Product:
  It is necessary to clearly define the relevant type of Product availability, the calculation method, and the equipment which significantly reduce availability. Request data/information from operators and/or maintainers.

- Are there any serious support problems relating to:

- task frequency?
- significant deviations from the predicted values of supportability KPI (eg, task duration, planned and real lead times)?
- success rate of a maintenance task below the required level (eg, repair tasks fail due to a lack of competence of maintenance personnel)?

The ISSO activities generate a set of recommendations about how to optimize the support solution or, if required, how to modify the Product.

The customer and contractor must determine in detail the goals of the ISSO process with respect to support optimization. The customer usually indicates the focus of an ISSO process (eg, the main goal of the ISSO process is cost reduction). It is recommended that customer and contractor prepare an ISSO Guidance Document (ISSO GD) to determine the detailed ISSO goals and the way to monitor and verify such goals.

2.2.3 The ISSO analysis object

ISSO analysis activities focus on the analysis object **task** in the following context:

- corrective maintenance tasks in terms of task performance problems and relevant supportability KPI
- operational support tasks in terms of task performance problems and relevant supportability KPI, including servicing activities
- preventive maintenance tasks in terms of task performance problems and relevant supportability KPI. The ISSO analysis process does not include all aspects already covered by the ISMO process of S4000P.
- disposal tasks
- software/data loading tasks in terms of maintenance or operational support activities

Note

Typically, the industry (both, integrator and supplier companies) is the first to perform supportability analysis activities to accompany the design and development process. Analysis like FMEA or PMA are often complex and need a deep technical knowledge about the Product and/or equipment/components installed in the Product. For this reason, it is not possible to expect feedback on this type of analysis from the operator. However, the experience gained by the maintainers/operators in performing daily support activities to the Product, and the support data collected systematically provide all necessary information that allows Product manufacturers or suppliers to perform ISSO analysis activities (same principle as for the ISMO process).

The ISSO analysis effort must be limited to a reasonable extent. For this purpose, it is strongly recommended to preselect the tasks to be analyzed. Those tasks are called the ISSO candidates. For the selection of analysis candidates, it is necessary to define the criteria and establish a candidate selection process. Typically, ISSO candidates are selected from the rectifying tasks, which usually resolve a task requirement (refer to Chap 12). However, supporting tasks can also be relevant for ISSO analysis. For example, a complex installation procedure for an equipment can be included in the ISSO analysis, whereas the referencing replace procedure is not relevant, because there are no problems with removal, test after installation, gain access or exit. However, as an ISSO principle, the supporting tasks will be analyzed in the context of the referencing rectifying task.

It is important to consider all ISSO analysis results for supporting tasks carefully within the referencing rectifying tasks.

2.2.4 Data collection for ISSO analysis

The success of an ISSO analysis is dependent on the possibility and on the willingness of the customer/operator to provide feedback to industry or to suppliers. This feedback derives from the experience of the operators/maintainers collected and documented during Product operation and during the performance of maintenance and operational support tasks. Documenting the

selected KPI and other data/information relevant for supportability are key to enabling a comparison between the predicted and the current situation, or to evaluating problems in support task performance.

Each type of task will have a corresponding set of questions for the ISSO analysis phase. If no data and/or information is available from the in-service use and support, in most cases it will be impossible to perform an ISSO process, because the in-service information is crucial to answer the questions related to the different types of tasks. Also refer to Para 3.3.

Note

It is recommended to consider the use and support feedback as part of an overall process to support risk and opportunity management.

To determine the amount and accuracy of supportability information obtained from the Product in its operational environment, it is advisable to make use of the available standard reporting systems of the customer. It is necessary to identify any shortfalls in measuring the achievements against the supportability goals established for the Product, and to verify supportability factors not tested during the procurement phases of the Product's life cycle. Additionally, it must be ensured to obtain the required field supportability data, which are not available through standard reporting systems.

For example, it is possible to implement a Data Reporting and Corrective Action System (DRACAS). DRACAS is a documented closed loop system for reporting, collecting, recording, analyzing, categorizing, investigating and taking timely effective corrective action on all discrepancies and failures relating to design, manufacturing and test processes. DRACAS can be viewed as a component function in the greater scheme of in-service IPS\LSA data. Even if there is a dedicated DRACAS system and it is relatively stand-alone, the data collected can become part of the overall data set beneficial for IPS. It is possible to integrate suitable opportunities for reporting into the usual maintenance practices. Facilities for reporting problems can be part of the following systems:

- maintenance scheduling and maintenance recording systems
- Interactive Electronic Technical Publications (IETP)

Whenever possible, it is recommended to use electronic Health and Usage Monitoring Systems (HUMS) and Data Loggers (DL) to reduce the risk of human errors.

Note

A DRACAS strategy can affect the Product design and the DRACAS/HUMS/DL technical requirement specification, as well as the selection process for an adequate Product with a proper support system.

Upon discovering or rectifying the problem, the maintenance personnel collects the data. It is preferable to use an approach that avoids multiple reviews or approvals of the reported problems. The reporting person must make data traceable, to verify information or obtain more details. The detailed approach and the IT solution implemented for data collection must be as user-friendly as possible. Whenever integrated into other systems, the data recording elements usually consider the context and avoid re-entering data. For example, when reporting a new problem encountered during a maintenance task, the option to report the problem must provide as much information as possible.

Additionally, the in-service data collection process will include, but will not be limited to:

- type of data
- actual date and update frequency
- exchange method
- need for confidentiality (especially in a military environment)
- intellectual property rights

The performance of a trade-off analysis between cost, data collection frequency and number of operational units to collect data from must aim to identify the best data collection plan, and ensure statistical relevance. To support the provision of the capability through the in-service phase, it is necessary to maintain supportability related data/information, which must include at least:

- Product use (eg, operating hours, distances)
- operating conditions
- occurrence of unknown symptoms leading to new troubleshooting procedures
- unexpected failures
- task frequency for corrective maintenance and operational support tasks
- performance of preventive and corrective maintenance activities (including problem reports, duration, lead times, success rate, personnel inadequacy)
- consumption of spares and consumables
- periods of unavailability due to maintenance activities
- capacity utilization of maintenance facilities
- cost factors

This determines the extent of in-service achievement of reliability and maintainability requirements, and identifies any shortfalls in the support solution. Analyzing these data will clarify the cause of shortfalls and, where appropriate, develop solutions and modifications for implementation.

All available information can be used for the ISSO process. During re-evaluation, it is important to document the reasons and assumptions made during the initial analyses and subsequent changes to be able to trace their origin. This includes, but is not limited to, Product/system descriptions, safety analyses, and requirements from all sources, trails reports, any form of user feedback, maintainer and user interviews, any changes or adaptations to technical documentation, number of man-hours, price and cost evaluations, obsolescence notices and expectance reports, and configuration baseline information.

Any product support solution should have a process in place to collect ideas from all layers of the involved organizations to optimize the solution and provide feedback to the persons who submitted the idea, thus ensuring a continuous process. The ISSO process must include these ideas.

Note

As an option, S5000F provides a specification to identify, document and exchange suitable feedback data and/or information. The data/information subject to documentation must be tailored to the concrete needs of the ISSO process. As an option, it is possible to combine the ISSO data/information requirements with other in-service feedback requirements to support further in-service analysis processes, for example ISMO, Failure Reporting and Corrective Action System (FRACAS) or Life Cycle Cost (LCC) analysis.

3 ISSO process

3.1 General assumptions

Several aspects determine the best way to establish an ISSO process, for example:

- the type of Product
- fleet size
- different use scenarios
- available data/information on the operator's side
- data/information quality
- required resources (personnel and budget) to perform the ISSO process
- chances for improvement

It is necessary to consider the costs and expected effectiveness of the ISSO process before defining the process. The final decision must also include the specific phase of life of the Product. The support solution developed for a new system is likely to be based on a set of assumptions and estimations, and the expected life of the system is long. A Product that has been in use for some time is likely to have many deviations from the original support concept because of pragmatic alternative solutions based on the experience gained by the support organization. A Product that is reaching the end of its life is likely to have a well-established support solution. The optimization effort must match the expected benefits.

Tracking KPI and reporting support problems ensure an ongoing optimization. The main driver of this optimization is the detection of deviations from previous figures, for example unexpected differences in spares consumption, man-hours or failure data. This requires an in-depth knowledge of the Product, its use and operating environment, as well as detailed information on the origin and calculation method of the figures used for tracking. The fleet size affects the statistical relevance of the figures and must be considered when establishing the calculation methods.

Various situations can make it necessary to re-evaluate the support solution, thus prompting a recurrent optimization process. The initial assumptions behind the decision are likely to have changed (eg, new use scenario, obsolete support resources on operator side, modified maintenance concept).

The optimization after hardware or software design changes can require the optimization of the support solution, too. This effort must be integrated in the design process, where the existing support solution is taken into consideration when designing the new solution. The process follows the regular IPS process, with the exception of additional design constraints coming from the existing in-service support solution.

3.2 ISSO phases

The ISSO process consists of three phases:

- Preparation phase:
  Definition of the framework for the ISSO effort and selection of ISSO candidates.
- Analysis phase:
  In-depth analysis of ISSO candidates, and development of recommendations for optimization.
- Follow-up phase:
  Together with the customer, evaluation of the recommendations generated during the analysis phase and recording of the final decision. If required, further activities will be defined and specified in order to contribute to the ongoing monitoring of the support solution, which will be implemented as a result of the ISSO process.

3.3 Preparation phase

3.3.1 Scope of the preparation phase

The first step of the ISSO process defines the scope of the optimization:

- Which Products/product variants are affected?
- Which maintenance and/or operational support tasks are affected?
- What aspects can be improved (eg, optimization of support costs, Product availability, reduction of resources, reduction of logistics footprint or development of skills)?

In addition, it is necessary to define a budget in terms of time and resources before commencing the work. The corresponding ISSO GD must include all methods used for optimization. When ordering the ISSO, the ISSO GD must be included in a contract, to ensure all stakeholders understand and agree on the activities.

Note

The document including the binding implementation rules for an ISSO process can be an ISSO GD or ISSO Process and Procedure Handbook (ISSO PPH). The term PPH is also well established, and is used for the implementation rules within S4000P, for example. Furthermore, the term ISSO GD is used.

After signing and agreeing on the ISSO GD, the stakeholders use the selected goals to organize the work. As a starting point, it is recommended to rank the items in such a way as to maximize the effect on the overall Product support solution (eg, top cost drivers or availability killers). This will reduce the risk of wasting resources. During the process, other possibilities of optimization can appear. Any possibility for optimization must be used, provided the required time and effort are taken into consideration. If the possible improvements have been rejected because of time or budget constraints, it is also necessary to include previously rejected ideas and then collected in an "idea box". When included in an overall update, the calculation can show different results.

Fig 2 illustrates the first part of the ISSO preparation phase. It is necessary to use this first phase to provide the general preconditions to start an ISSO process. For example, it does not make any sense to resume the ISSO activities if the ISSO GD is not available in any form (including as a draft that provides a basis for discussion).

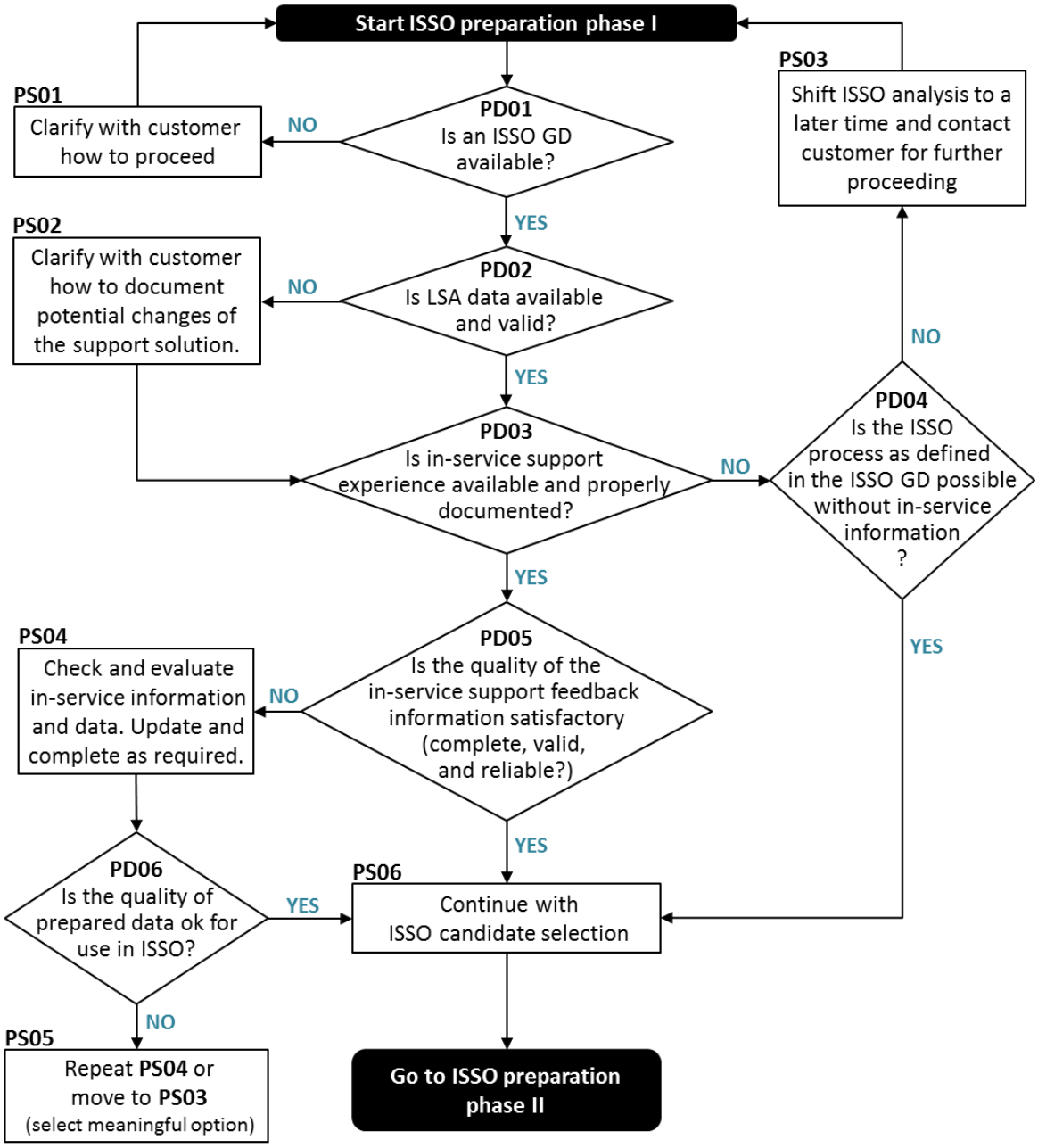
3.3.2 Preparation phase process logic

The following logic diagrams present the decisions and steps necessary to set up a proper ISSO process. After all decisions are taken and the preconditions to start the ISSO analysis are met, specific analysis activities for the tasks selected as ISSO candidates can start.

The logic diagrams show the individual process steps (S) and decisions (D), numbered consecutively. Each process step and decision has an alphanumeric code **PXYY**, where:

- Digit 1 : **P** Code for the ISSO preparation phase (always "**P**" for **P**reparation)
- Digit 2 : **X** Code for the type of logic diagram element ("**S**"= step, "**D**"= decision)
- Digit 3,4 : **YY** Number, ascending from 01 to 99

Table 2 and Table 3 provide details on the process steps and decisions.



ICN-B6865-S3000L0130-001-01

Fig 2 ISSO preparation phase I

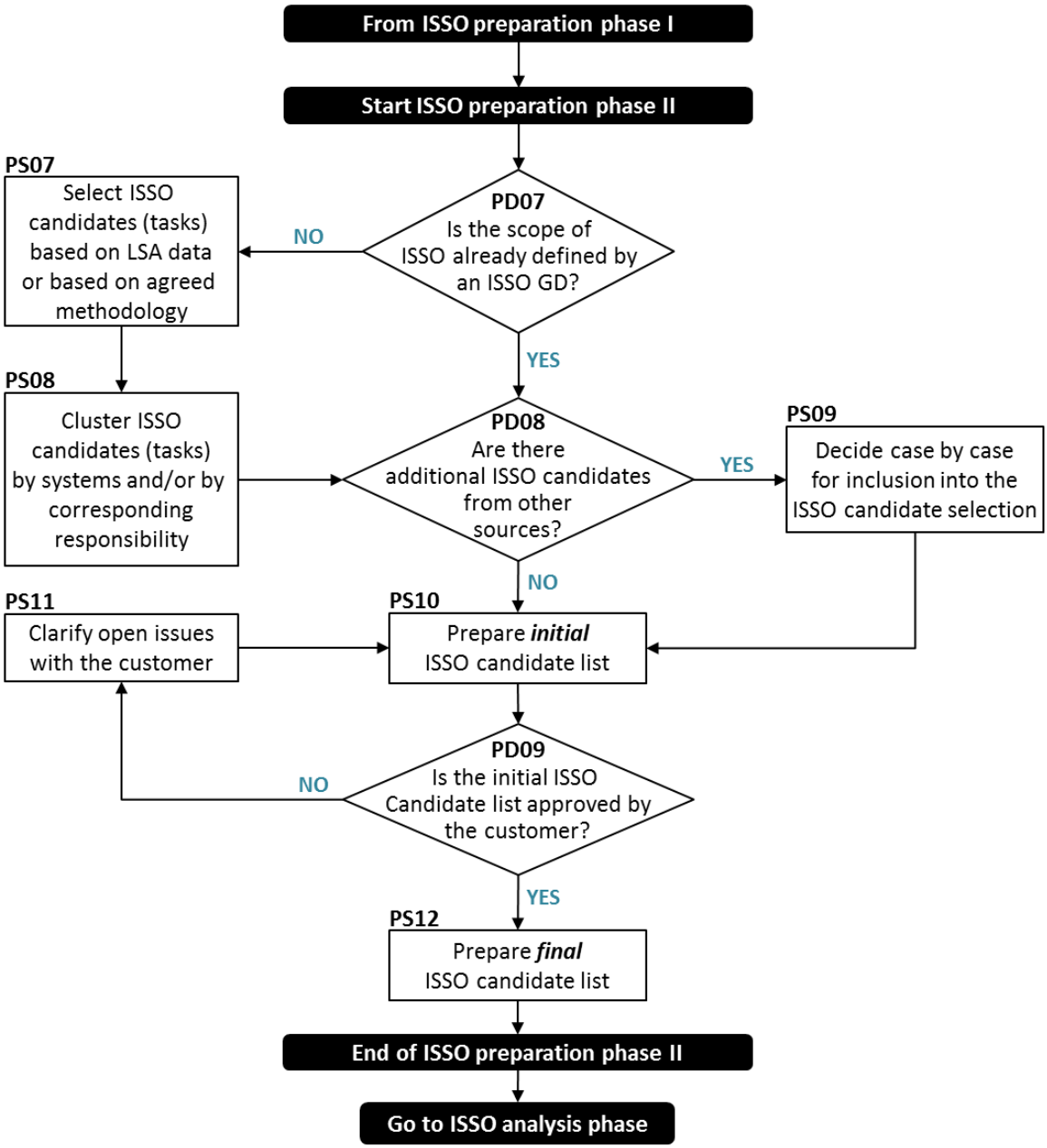
When all preconditions are met to a sufficient extent, it is possible to close ISSO preparation phase I. Table 2 illustrates all the decisions and steps included in this phase.

Table 2 Process steps and decisions from ISSO preparation phase I

Code	Description
PD01	An ISSO GD is crucial for an ISSO program. After establishing clear implementation rules, it is possible to start the process.

Code	Description
PS01	The customer and contractor must define, agree to and sign clear implementation rules. If this precondition is not given, the customer must indicate how to proceed.
PD02	LSA data is a source of information useful in selecting the ISSO candidates and documenting the results of the ISSO program. This includes documenting all required support solution changes within the LSA data, and coordinating all the corresponding changes required within the IPS elements.
PS02	If no LSA data is available, it is necessary to determine an alternative methodology to document the changes of the support solution. LSA data is a perfect option, because LSA data typically include information about, for example, Product breakdown, task requirements and support tasks (including the complete MTA).
PD03	Determining the extent of the in-service experience earned is vital to take decisions. It is of crucial relevance to make available any piece of information that can help evaluate any issues about the existing maintenance or operational support tasks. This type of information is necessary to analyze everything that has occurred during repair activities, operational support and Product servicing, as well as to analyze problems or unexpected deviations of the KPI detected by the maintainer or operator. In addition to databases or tools populated during the Product operation and support, the experience of the operators/maintainers provides a lot of valuable information. It is necessary to consider this source of information during this phase, to interview all relevant personnel and document the result of the interviews carefully. It is strongly recommended to use this input during the ISSO candidate selection and the ISSO analysis phase.
PD04	It is necessary to decide whether to continue the ISSO process despite non-existent or very limited in-service data/information. In most cases, it is better to postpone the ISSO cycle until a better information basis is available.
PS03	Consequence of unavailable in-service data/information.
PD05	It is necessary to evaluate the available in-service data/information to ensure the best possible quality and validity. A large collection of data can be useless if the data it contains is outdated, unreliable, incomplete, or invalid with respect to the last operating conditions of the Product, last Product configuration, etc.
PS04	If available in-service data/information seem insufficient, it is necessary to seek additional information sources or possibilities to update/improve the existing data/information.
PD06	Evaluate the results of PS04. If the updated in-service data/information show a significant improvement, and if the currently available data/information are suitable for an ISSO program, continue with the ISSO candidate selection as the second part of the ISSO preparation phase. Otherwise, go to PS05.
PS05	The update/improvement of the in-service data was not successful. In this case, it is necessary to analyze the data again. This determines whether it makes sense to start the ISSO program with the available data/information, bearing in mind that the conditions are not ideal. Try to activate additional information sources. However, if it is not possible to establish a satisfying data/information basis, move to PS03 and postpone the ISSO program.
PS06	Go to ISSO preparation phase II for the ISSO candidate selection.

After the ISSO preparation phase I, the scope concerning the ISSO candidates is determined, as illustrated in Fig 3. Table 3 illustrates all the decisions and steps included in ISSO preparation phase II.



ICN-B6865-S3000L0131-001-01

Fig 3 ISSO preparation phase II

Table 3 Process steps and decisions from ISSO preparation phase II

Code	Description
PD07	If the ISSO GD illustrates the scope of the ISSO program in detail, it is not necessary to go through a detailed selection process based on defined criteria for the tasks.

Code	Description
PS07	Based on the actual support solution, it is necessary to include all potential support activities in the ISSO candidate selection process. The criteria justifying when a task becomes an ISSO candidate will be applied to each task. The result will be a list of tasks relevant for the ISSO analysis phase. The ISSO candidate selection is limited to rectifying tasks only. The reason for this approach is the structuring of tasks given in Chap 12. Rectifying tasks (eg, a replace task) reference the supporting tasks (eg, a removal of an equipment) as subtasks of the rectifying tasks. Normally, all supporting tasks documented in the LSA data are referenced at least by one rectifying task. Consequently, the ISSO analysis automatically includes all supporting tasks referenced by rectifying tasks selected as ISSO candidates.
PS08	If required, it is necessary to prepare the results of PS07 to assign the analysis activities to the different stakeholders involved. For example, the different companies taking part in a project can bear the responsibility of the data. Especially in multi-national projects, it is necessary to define clearly who is responsible for LSA data, technical publication, spare part supply, or training. In the context of the complete LSA/IPS scope, the same applies to the ISSO program.
PD08	Are there any additional tasks not documented in the LSA data or in the IPS elements, but which were performed, for example, based on individual decisions made by the operators/maintainers? If yes, it is necessary to determine whether those activities can also become an ISSO candidate, which must be analyzed within an ISSO program, especially if those activities caused problems.
PS09	Determine additional tasks to include in the ISSO program.
PS10	With the results from the ISSO candidate selection from PS07/PS08, combined with the additional identified ISSO candidates from PS09, prepare an initial ISSO candidate list including all identified tasks to be analyzed.
PD09	The customer must receive and approve the initial ISSO candidate list. If not approved, continue with PS11.
PS11	If the customer has any questions on the initial ISSO candidate list, clarify them before the customer and contractor determine the final ISSO candidate list.
PS12	After the customer and contractor reached an agreement, prepare the final ISSO candidate list, which constitutes the binding baseline for the ISSO program.

3.4 Analysis phase

3.4.1 Scope of the analysis phase

Within the ISSO analysis phase, the maintenance and operational support tasks selected as ISSO candidates are analyzed against a set of criteria. To ensure the best possible analysis result, there is a classification of tasks to assign the most suitable analysis criteria to each type of task. For each ISSO candidate, the corresponding classification leads to the specific "box of criteria".

Fig 4 illustrates the overall logic diagram for the complete ISSO analysis phase. As the result of the ISSO preparation phase, the list of the ISSO candidates serves as the input to the ISSO analysis phase. First, when starting the analysis, it is necessary to determine the correct analysis block (marked from **A** to **G**) for the task. Table 4 illustrates all the decisions and steps included in this phase.

The general classification of the ISSO candidates comprises the following:

- corrective maintenance tasks (eg, repair, replace)

- servicing tasks
- test, fault location or inspection tasks
- preventive maintenance packages (including single tasks within the package)
- PHST tasks
- disposal tasks
- software or data loading tasks

3.4.2 Analysis phase process logic

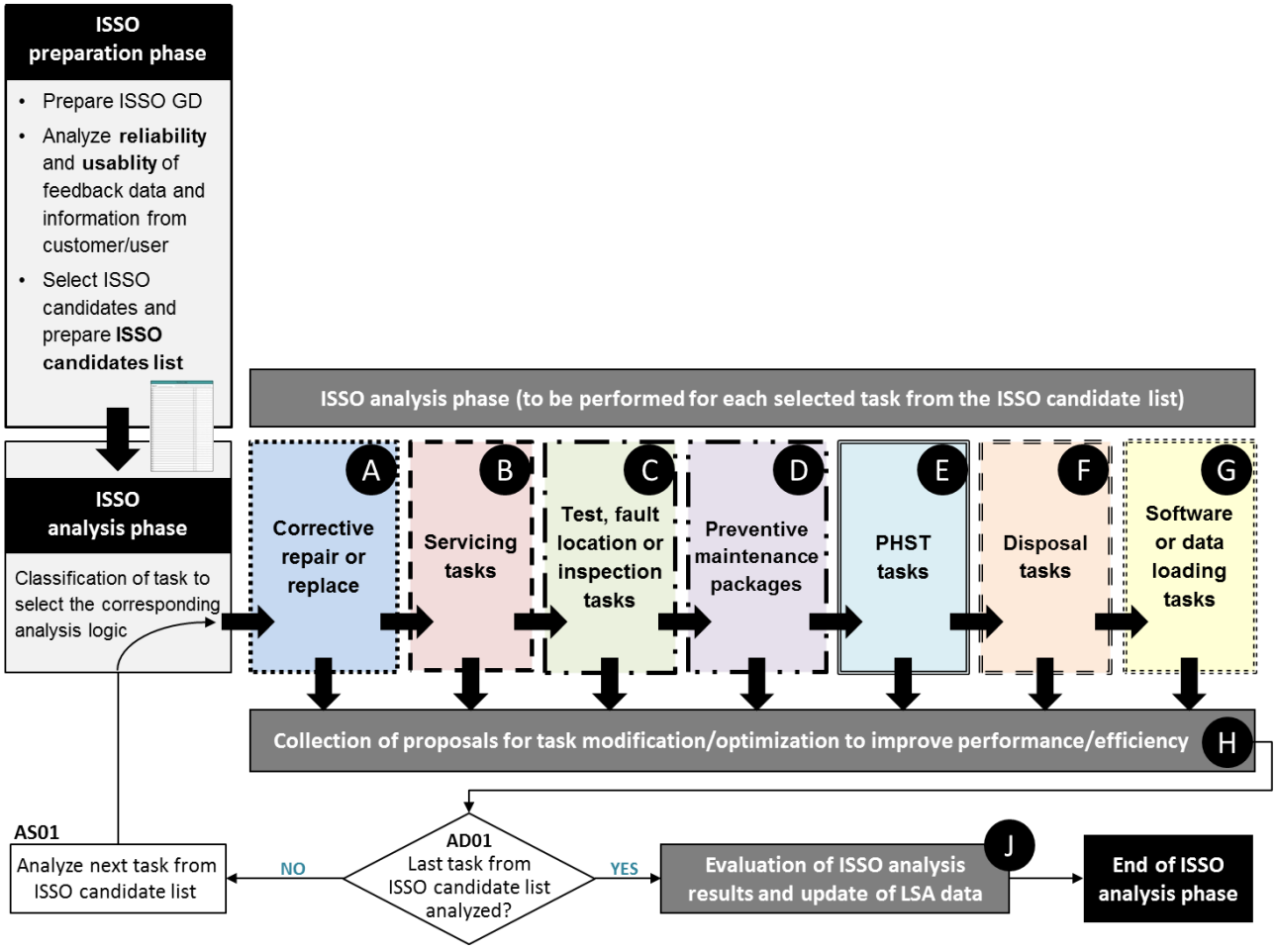
The following logic diagrams (from Fig 4 to Fig 11) show the decisions and steps, which are necessary to perform the required analysis activities. Table 4 to Table 11 illustrate all the decisions and steps.

The logic diagrams show the individual process steps (S) and decisions (D) numbered consecutively. Each process step and decision has an alphanumeric code **PXYY**, where:

- Digit 1 : **P** Code for the ISSO analysis phase (always "**A**" for **A**nalysis)
- Digit 2 : **X** Code for the type of logic diagram element ("**S**"= step, "**D**"= decision)
- Digit 3,4 : **YY** Number, ascending from 01 to 99

3.4.3 Overall analysis logic diagram

The ISSO candidates belonging to different rectifying tasks are clustered in different analysis areas. First, it is necessary to determine to which analysis block the task belongs. The single logic diagrams include this decision for the different task classes.



ICN-B6865-S3000L0132-001-01

Fig 4 Overall logic diagram of the ISSO analysis phase

At the end of each ISSO analysis block, it is necessary to document the analysis result for each analyzed task (block **H**). The result can be a proposal on how to resolve the identified or reported problems, for example by changing the analyzed task or suggesting that there is no need for any change. If required, it is also possible to include an explanation for any proposal. After block H, the analysis cycle for the next task from the ISSO candidate list starts. Finally, all ISSO results will be evaluated to decide whether to implement the optimization proposals. For all agreed changes, LSA data must be updated accordingly (block **J**).

Table 4 Process steps and decisions from overall ISSO analysis logic

Code	Description
AD01	Determine the end of the analysis phase following the analysis of the last task from the ISSO candidate list.
AS01	Continue with the next task from the ISSO candidate list, classify the task in order to choose the appropriate analysis block (**A** to **G**).

3.4.4 Detailed analysis logic diagrams for different task classes

The general ISSO analysis flowcharts which are presented in the following paragraphs should be considered as the baseline for an individual ISSO process dependent on project/Product specific requirements, for example ISSO triggers, specific assumptions for the ISSO program given by the customer, or specific ISSO goals determined by the customer. The general logic diagrams need to be detailed, complemented or tailored as required. The ISSO GD must document the detailed logic diagrams that can be significantly extended. Moreover, the customer and the contractor who performs the ISSO program must harmonize and agree on the detailed logic diagrams in the ISSO GD. Para 3.4.5 shows an example of detailed logic diagram for one of the task classes. The relevant regulatory authority must harmonize and agree on any aspects, which must be considered in the context of Product certification.

3.4.4.1 ISSO analysis logic diagram for corrective maintenance tasks

The following logic diagram represents the ISSO analysis baseline for corrective maintenance tasks:

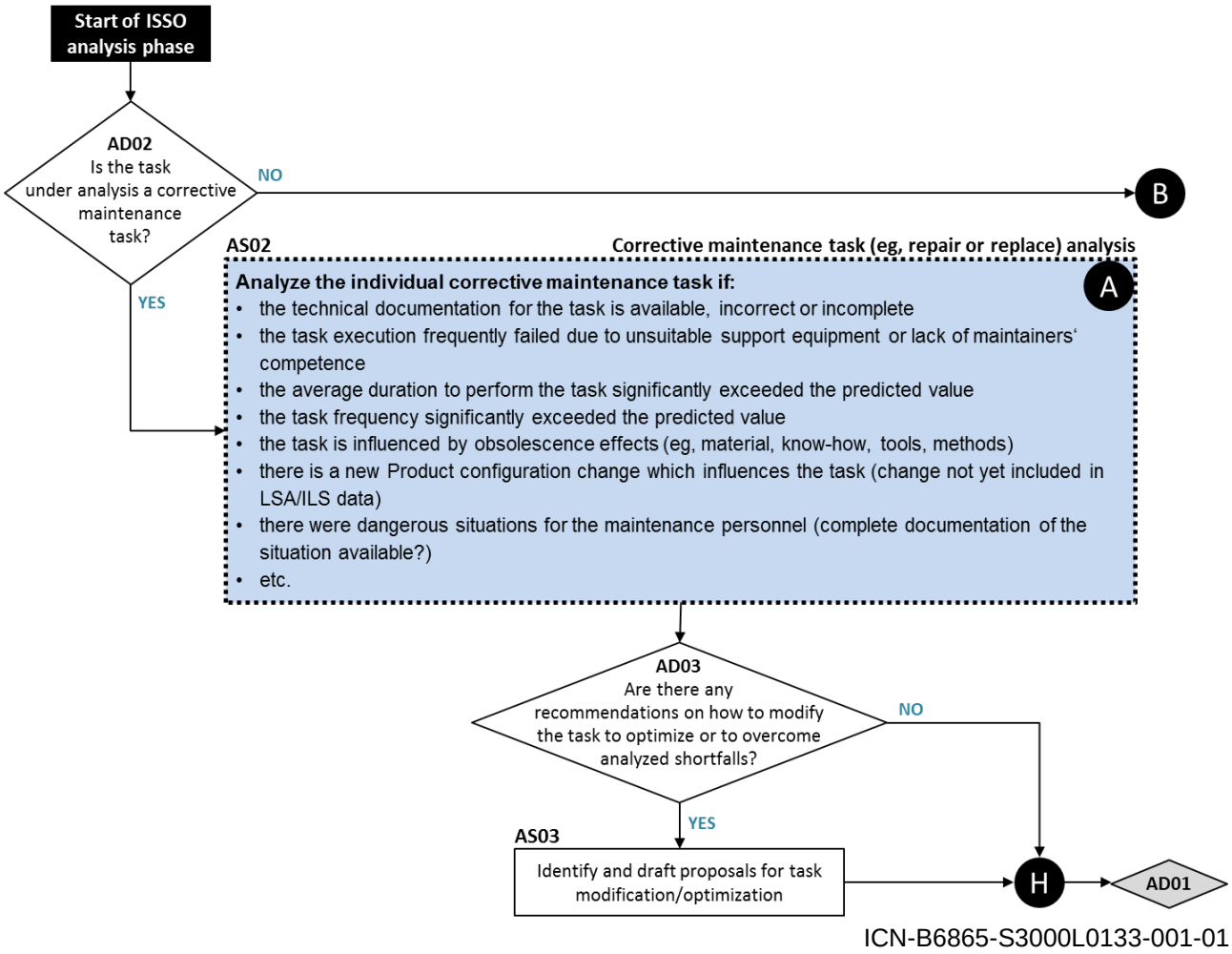


Fig 5 ISSO analysis logic diagram for corrective maintenance tasks

Table 5 Process steps and decisions for corrective maintenance tasks

Code	Description
AD02	Decide to start performing the activities of the analysis block for the corrective maintenance tasks
AS02	It includes a list of potential analysis questions for corrective maintenance tasks. The list can be complemented or tailored as required. The questions must be arranged in a logic diagram, structured in an adequate sequence, preferably with the indication of potential interdependencies.
AD03	Determine whether the corrective maintenance task under analysis requires any changes/optimization (or other proposals).
AS03	If there are any recommendations about how to modify/optimize the corrective maintenance task based on the analysis steps performed in block **A**, draft and justify the proposals, as required.
AD01	After completing the analysis and documenting its result (block **H**), return to the overall logic diagram to continue with the next task or to finish the ISSO analysis phase, if all tasks of the ISSO candidate list have been analyzed.

Note

A corrective maintenance task is not limited to repair or replace activities. Also, other activities, (eg, calibrations, adjustments or lubrication) can be classified as corrective maintenance tasks.

3.4.4.2 ISSO analysis logic diagram for servicing tasks

The following logic diagram represents the ISSO analysis baseline for servicing tasks:

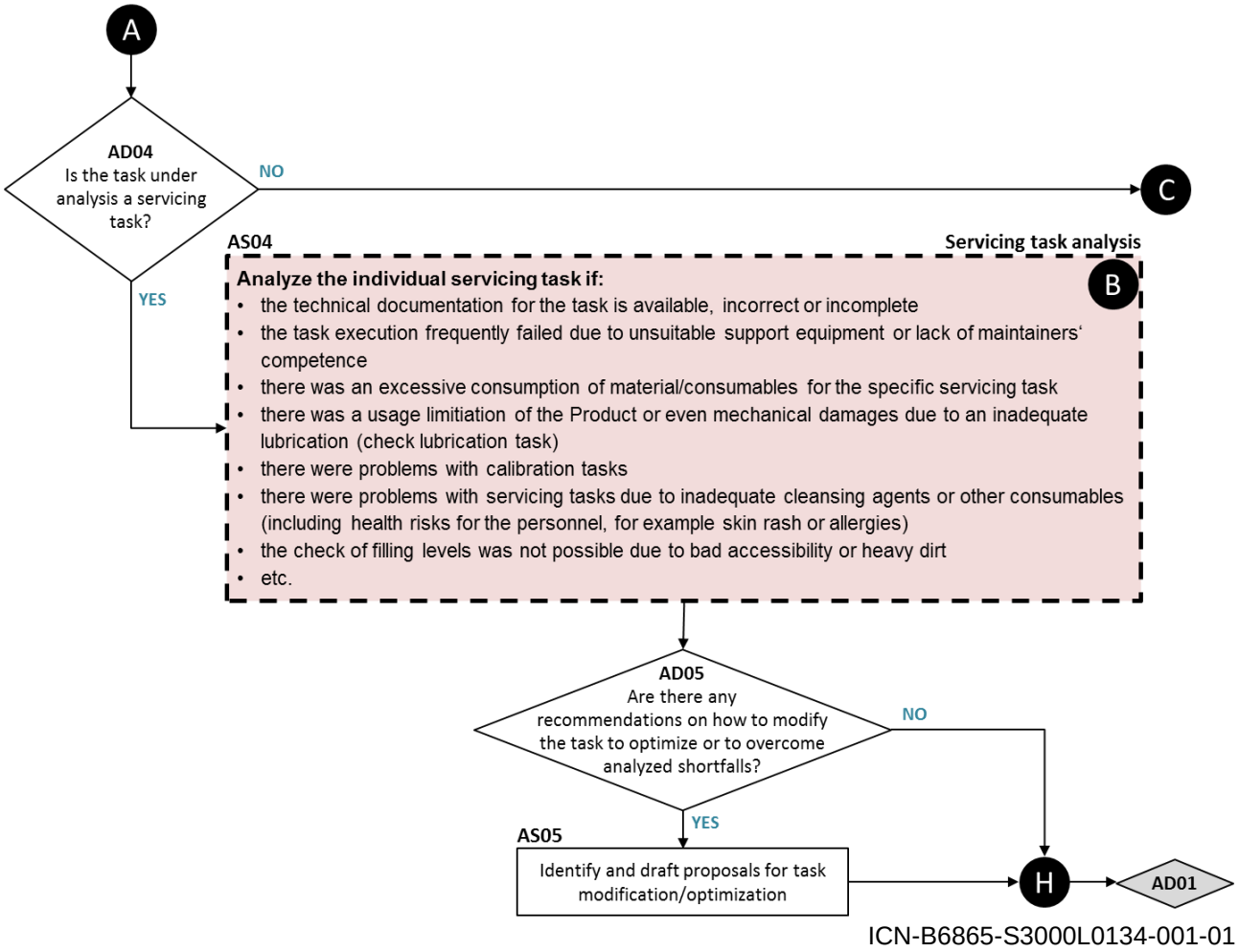


Fig 6 ISSO analysis logic diagram for servicing tasks

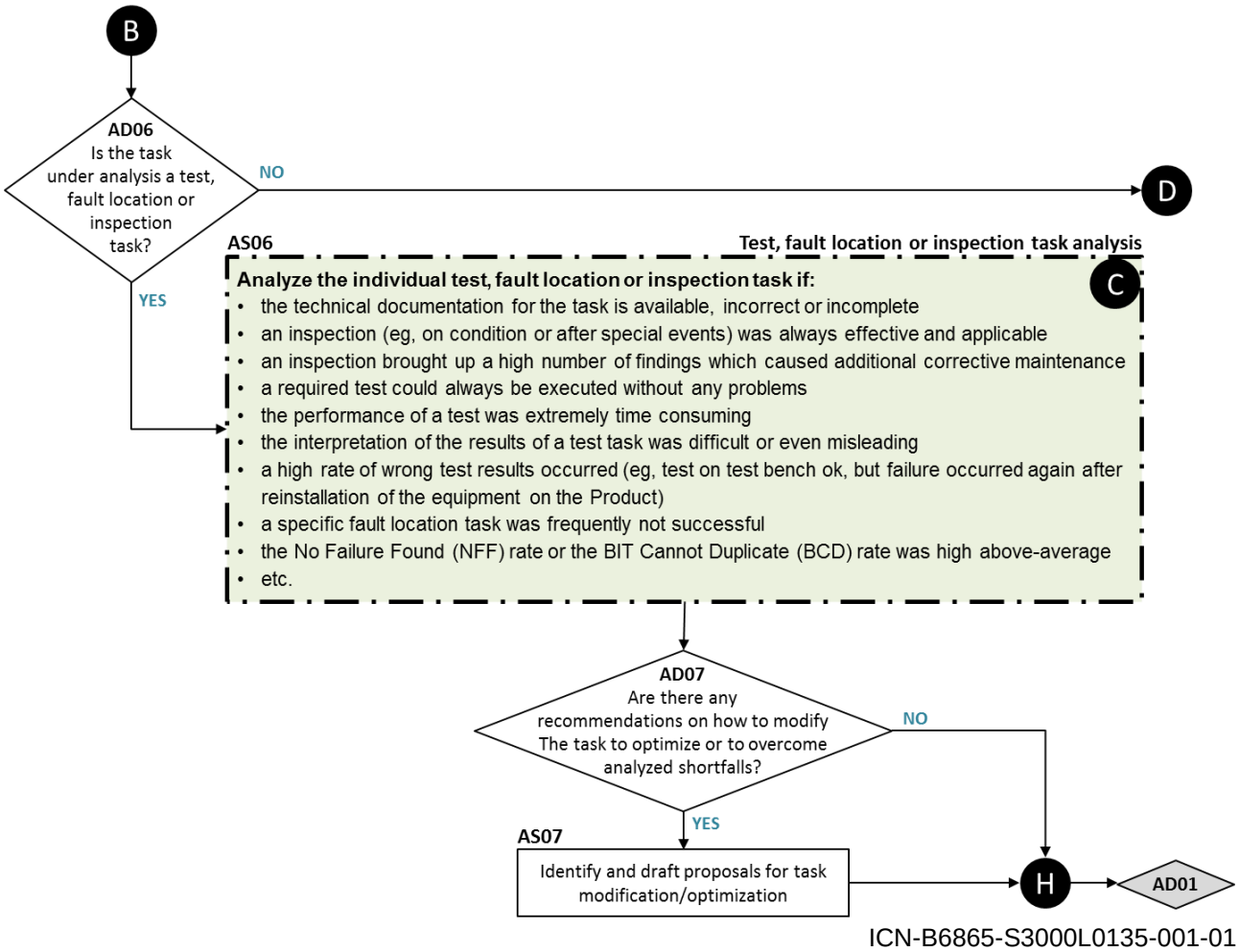
Table 6 Process steps and decisions for servicing tasks

Code	Description
AD04	Decide to start performing the activities of the analysis block for the servicing tasks.
AS04	It includes a list of potential analysis questions for servicing tasks. The list can be complemented or tailored as required. The questions must be arranged in a logic diagram, structured in an adequate sequence, preferably with the indication of potential interdependencies.
AD05	Determine whether the servicing task under analysis requires any changes/optimization (or other proposals).

Code	Description
AS05	If there are any recommendations about how to modify/optimize the servicing task based on the analysis steps performed in block **B**, draft and justify the proposals, as required.
AD01	After completing the analysis and documenting its result (block **H**), return to the overall logic diagram to continue with the next task or to finish the ISSO analysis phase, if all tasks of the ISSO candidate list have been analyzed.

3.4.4.3 ISSO analysis logic diagram for test, fault location or inspection tasks

The following logic diagram represents the baseline for ISSO analysis for test, fault location or inspection tasks:



ICN-B6865-S3000L0135-001-01

Fig 7 ISSO analysis logic diagram for test, fault location or inspection tasks

Table 7 Process steps and decisions for test, fault location or inspection tasks

Code	Description
AD06	Decide to start performing the activities of the analysis block for the test, fault location or inspection tasks.

Code	Description
AS06	It includes a list of potential analysis questions for test, fault location or inspection tasks. The list can be complemented or tailored as required. The questions must be arranged in a logic diagram, structured in an adequate sequence, preferably with the indication of potential interdependencies.
AD07	Determine whether the test, fault location or inspection task under analysis requires any changes/optimization (or other proposals).
AS07	If there are any recommendations about how to modify/optimize the test, fault location or inspection task based on the analysis steps performed in block **C**, draft and justify the proposals, as required.
AD01	After completing the analysis and documenting its result (block **H**), return to the overall logic diagram to continue with the next task or to finish the ISSO analysis phase, if all tasks from the ISSO candidate list have been analyzed.

3.4.4.4 ISSO analysis logic diagram for preventive maintenance packages

The following logic diagram represents the baseline for ISSO analysis for preventive maintenance packages:

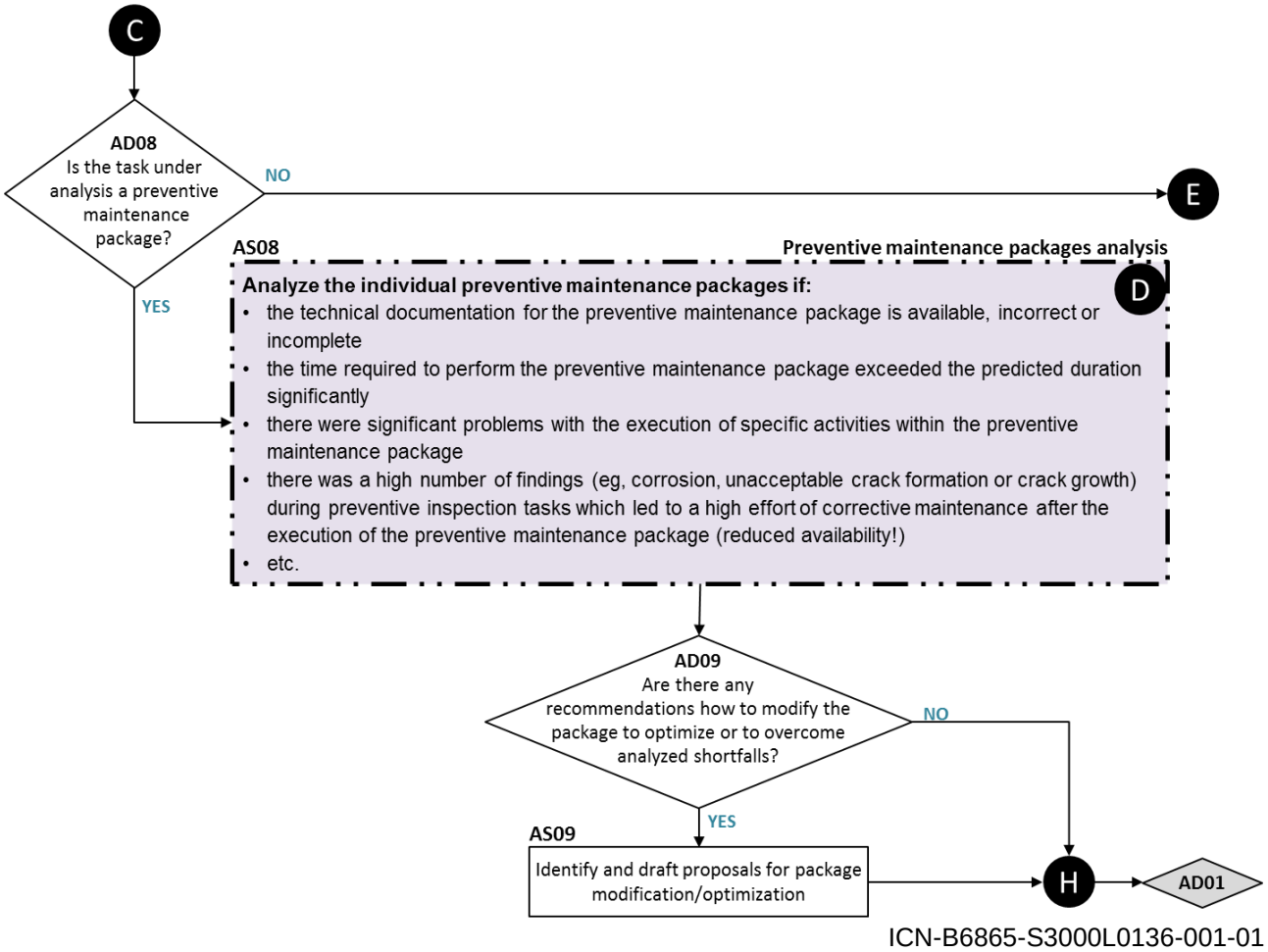


Fig 8 ISSO analysis logic diagram for preventive maintenance packages

Table 8 Process steps and decisions for preventive maintenance packages

Code	Description
AD08	Decide to start performing the activities of the analysis block for the preventive maintenance packages.
AS08	It includes a list of potential analysis questions for preventive maintenance packages. The list can be complemented or tailored as required. The questions must be arranged in a logic diagram, structured in an adequate sequence, preferably with the indication of potential interdependencies.
AD09	Determine whether the preventive maintenance package under analysis or activities within the preventive maintenance packages require any changes/optimization (or other proposals).
AS09	If there are any recommendations about how to modify/optimize the preventive maintenance package or activities within the preventive maintenance package based on the analysis steps performed in block **D**, draft and justify the proposals, as required.
AD01	After completing the analysis and documenting its result (block **H**), return to the overall logic diagram to continue with the next task or to finish the ISSO analysis phase, if all tasks from the ISSO candidate list have been analyzed.

3.4.4.5 ISSO analysis logic diagram for PHST tasks

The following logic diagram represents the baseline for ISSO analysis for PHST tasks:

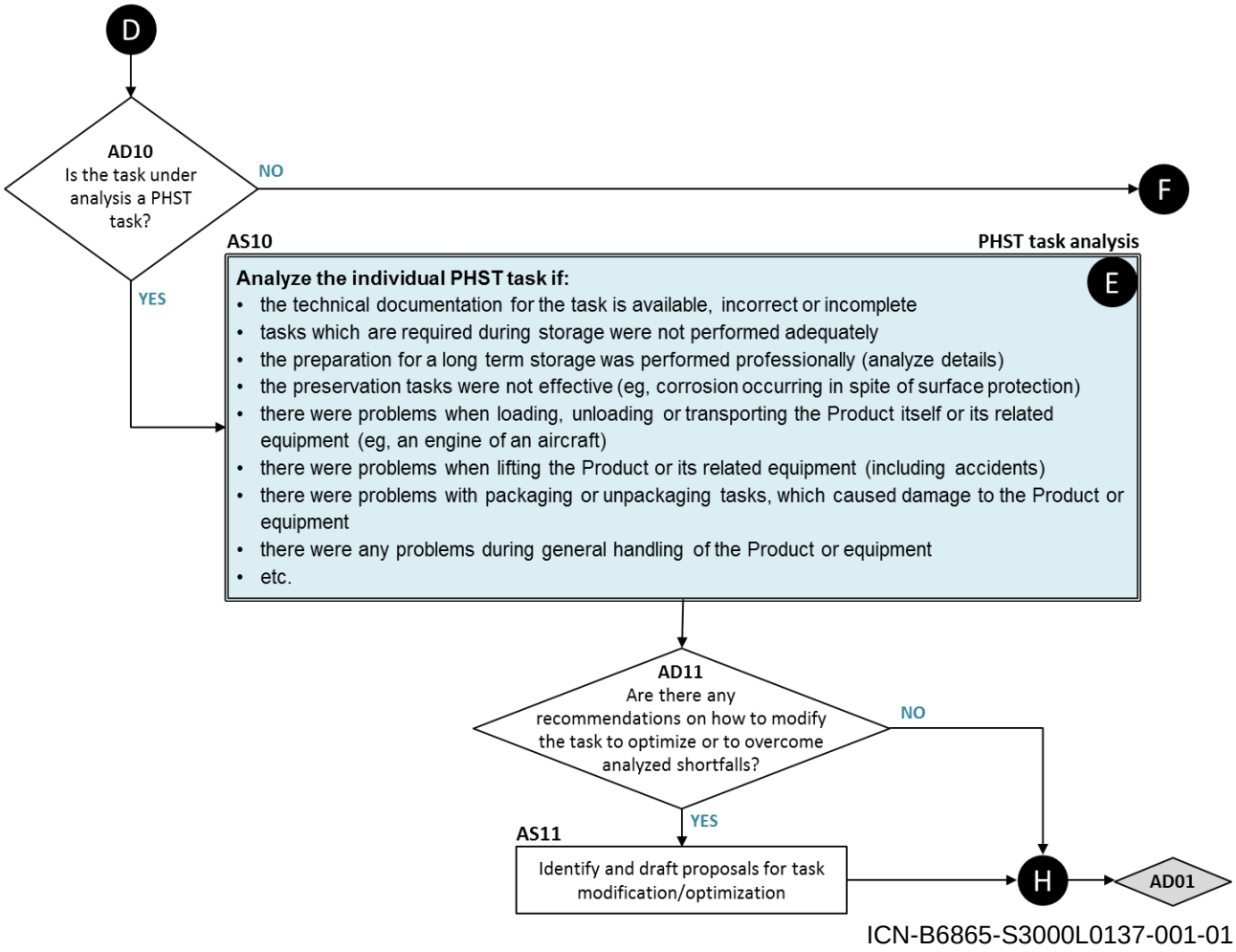


Fig 9 ISSO analysis logic diagram for PHST tasks

Table 9 Process steps and decisions for PHST tasks

Code	Description
AD10	Decide to start performing the activities of the analysis block for the PHST tasks.
AS10	It includes a list of potential analysis questions for PHST tasks. The list can be complemented or tailored as required. The questions must be arranged in a logic diagram, structured in an adequate sequence, preferably with the indication of potential interdependencies.
AD11	Determine whether the PHST task under analysis requires any changes/optimization (or other proposals).
AS11	If there are any recommendations about how to modify/optimize the PHST task based on the analysis steps performed in block **E**, draft and justify the proposals for the task under analysis, as required.
AD01	After completing the analysis and documenting its result (block **H**), return to the overall logic diagram to continue with the next task or to finish the ISSO analysis phase, if all tasks from the ISSO candidate list have been analyzed.

3.4.4.6 ISSO analysis logic diagram for disposal tasks

The following logic diagram represents the baseline for ISSO analysis of disposal tasks:

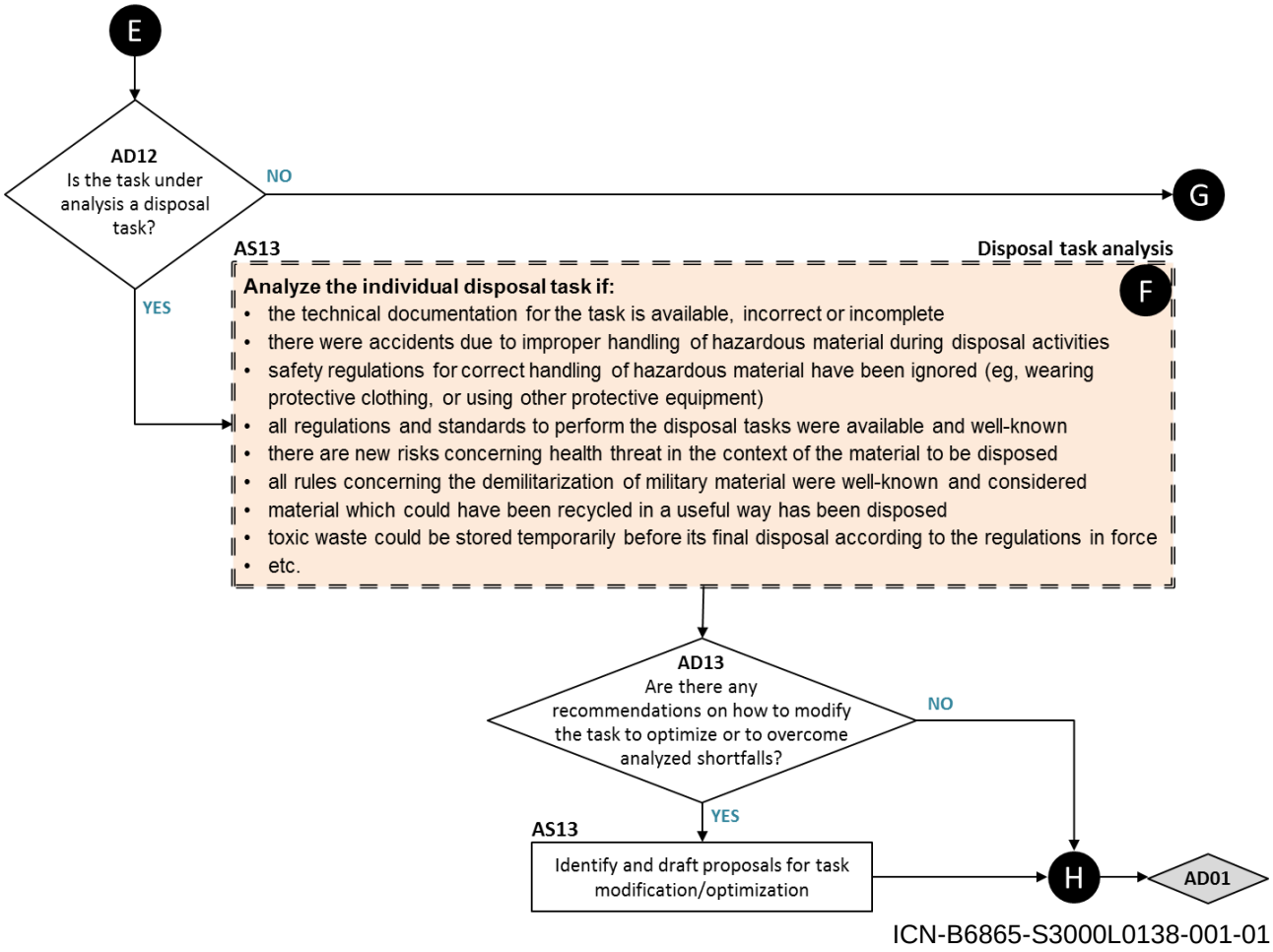


Fig 10 ISSO analysis logic diagram for disposal tasks

Table 10 Process steps and decisions for disposal tasks

Code	Description
AD12	Decide to start performing the activities of the analysis block for the disposal tasks.
AS12	It includes a list of potential analysis questions for disposal tasks. The list can be complemented or tailored as required. The questions must be arranged in a logic diagram, structured in an adequate sequence, preferably with the indication of potential interdependencies.
AD13	Determine whether the disposal task under analysis requires any changes/optimization (or other proposals).
AS13	If there are any recommendations about how to modify/optimize the disposal task based on the analysis steps performed in block **F**, draft and justify the proposals, as required.
AD01	After completing the analysis and documenting its result (block **H**), return to the overall logic diagram to continue with the next task or to finish the ISSO analysis phase, if all tasks from the ISSO candidate list have been analyzed.

3.4.4.7 ISSO analysis logic diagram for software or data loading tasks

The following logic diagram represents the baseline for ISSO analysis for software or data loading tasks:

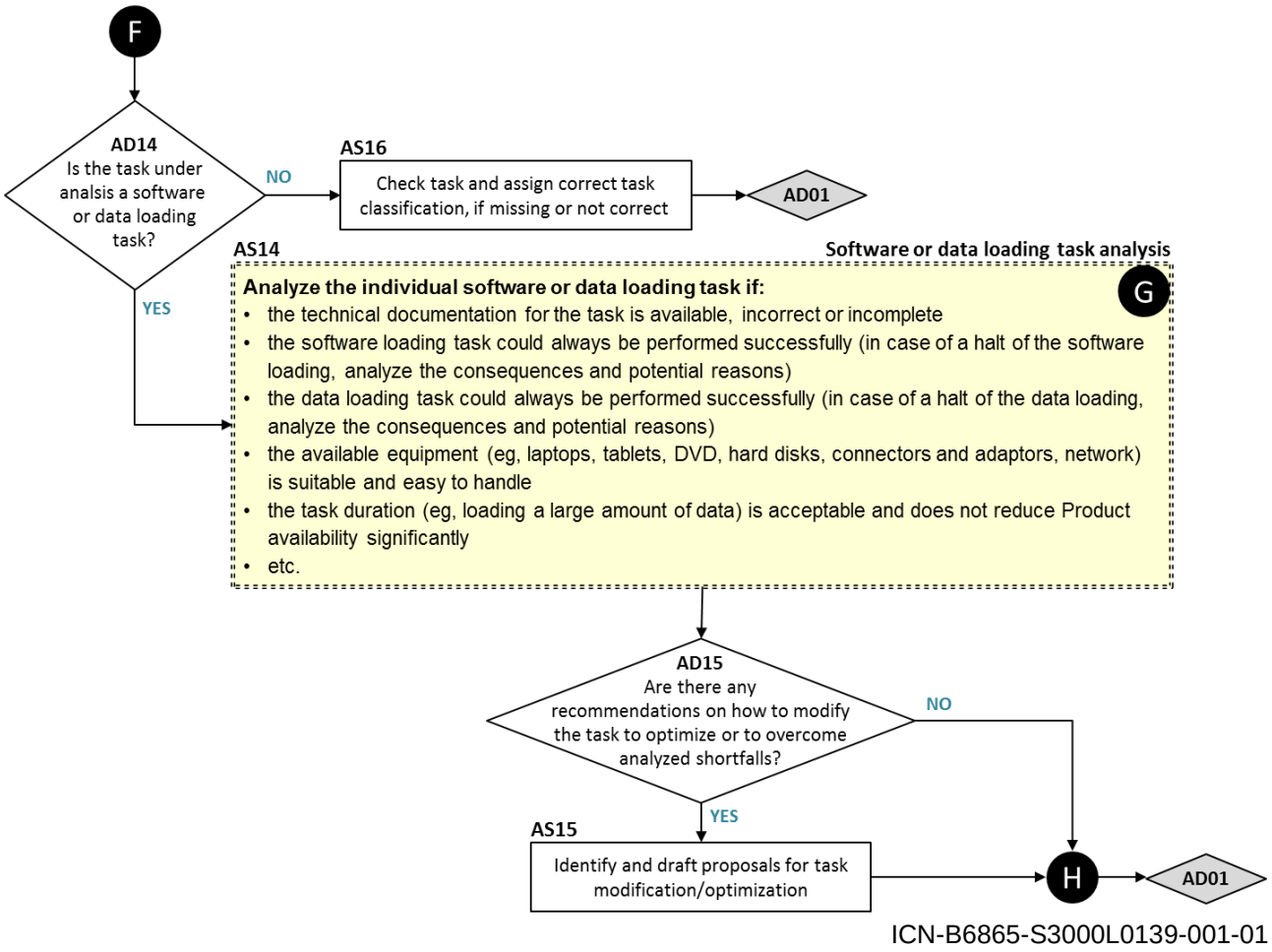


Fig 11 ISSO analysis logic diagram for software or data loading tasks

Table 11 Process steps and decisions for software or data loading tasks

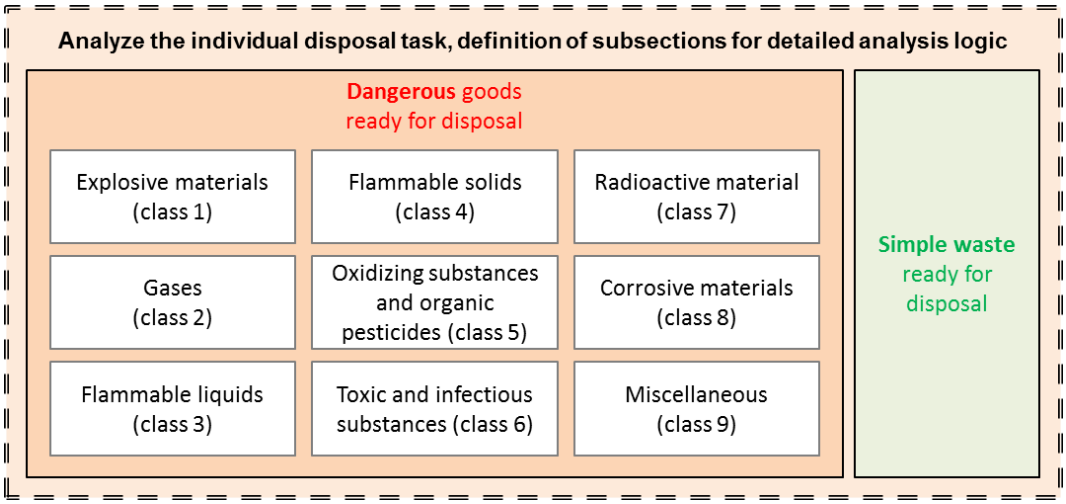
Code	Description
AD14	Decide to start performing the activities of the analysis block for software or data loading tasks.
AS14	It includes a list of potential analysis questions for software or data loading tasks. The list can be complemented or tailored as required. The questions must be arranged in a logic diagram, structured in an adequate sequence, preferably with the indication of potential interdependencies.
AD15	Determine whether the software or data loading task under analysis requires any changes/optimization (or other proposals).
AS15	If there are any recommendations about how to modify/optimize the software or data loading task based on the analysis steps performed in block **G**, draft and justify the proposals, as required.

Code	Description
AS16	If the answer to decision point AD14 is "NO", this means it was not possible to include the task from the ISSO candidate list in one of the given categories. It is then necessary to reassess the task and integrate the missing or change the misleading category, to be able to choose the right analysis block (refer to Fig 4).
AD01	After completing the analysis and documenting its result (block **H**), return to the overall logic diagram to continue with the next task or to finish the ISSO analysis phase, if all tasks from the ISSO candidate list have been analyzed.

3.4.5 Example for a detailed analysis logic

The generic analysis logics given in Para 3.4.4.1 to Para 3.4.4.7 are the baseline for the development of more detailed analysis logics, which typically depend on the specific Product and/or project. An ISSO GD must document those specific analysis logics. These logics are typically more complex, and experienced personnel must prepare and adapt them to the needs of the tasks included in the ISSO activities.

In the example below, the task category "disposal task" illustrates the difference between the generic logic in Para 3.4.4.6 and a detailed, project-specific logic. A first step is to provide a further classification of a disposal task based on the material that requires disposal (refer to Fig 12).



ICN-B6865-S3000L0140-001-01

Fig 12 Classification of disposal tasks to prepare detailed analysis logics

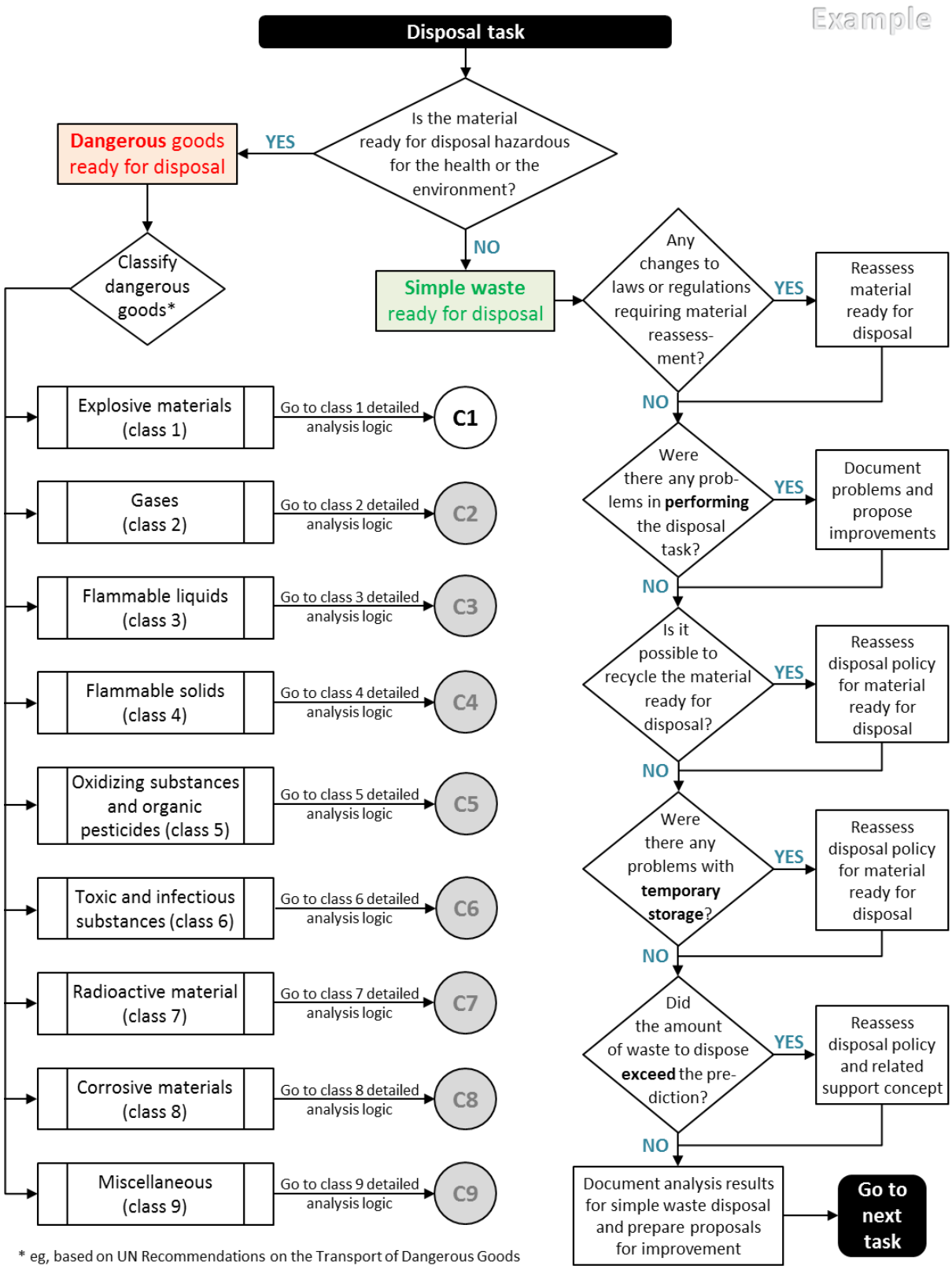
The classification of disposal tasks above allows the development of specific analysis logics for each material that needs to be disposed within the project.

The following logic flowcharts (Fig 13 and Fig 14) are examples to show a typical structure of a detailed analysis logic documented within an ISSO GD.

Fig 13 shows the subdivision of the disposal task. The first step is to distinguish between dangerous goods and "simple" waste. It is evident that these different types of material must answer to different analysis questions, which determine whether a task is efficient and applicable, and whether there can be problems during the task performance. There is a set of questions for the "Simple waste ready for disposal." Additionally, the different types of dangerous goods require different analysis logics. The different types of goods shown in the example are included the "UN Recommendations on the Transport of Dangerous Goods."

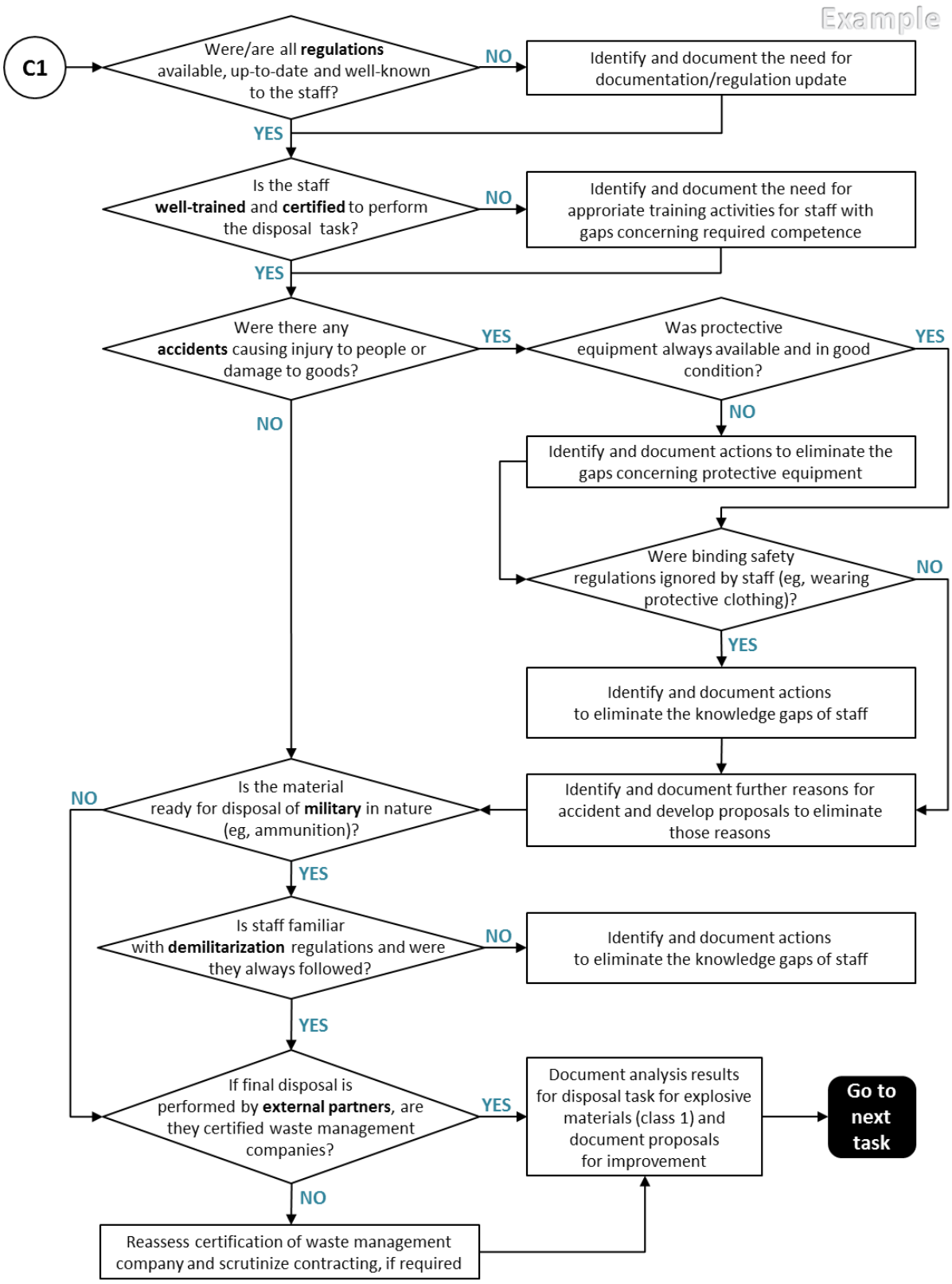
Fig 14 provides an example of a detailed disposal task logic for class 1 (explosive materials). It is possible to handle all other material classes in a similar way. The analysis logic can vary for each class, because each class has specific aspects that must be taken into consideration for the disposal of the specific material.

The result of the analysis must always be the same. Finally, there can be a recommendation not to change the task, there can be proposals to optimize/improve the task, or even the maintenance and support concept if the optimization/improvement of the concept proves to be useful.



ICN-B6865-S3000L0141-001-01

Fig 13 Overarching selection logic for disposal tasks analysis



ICN-B6865-S3000L0142-001-01

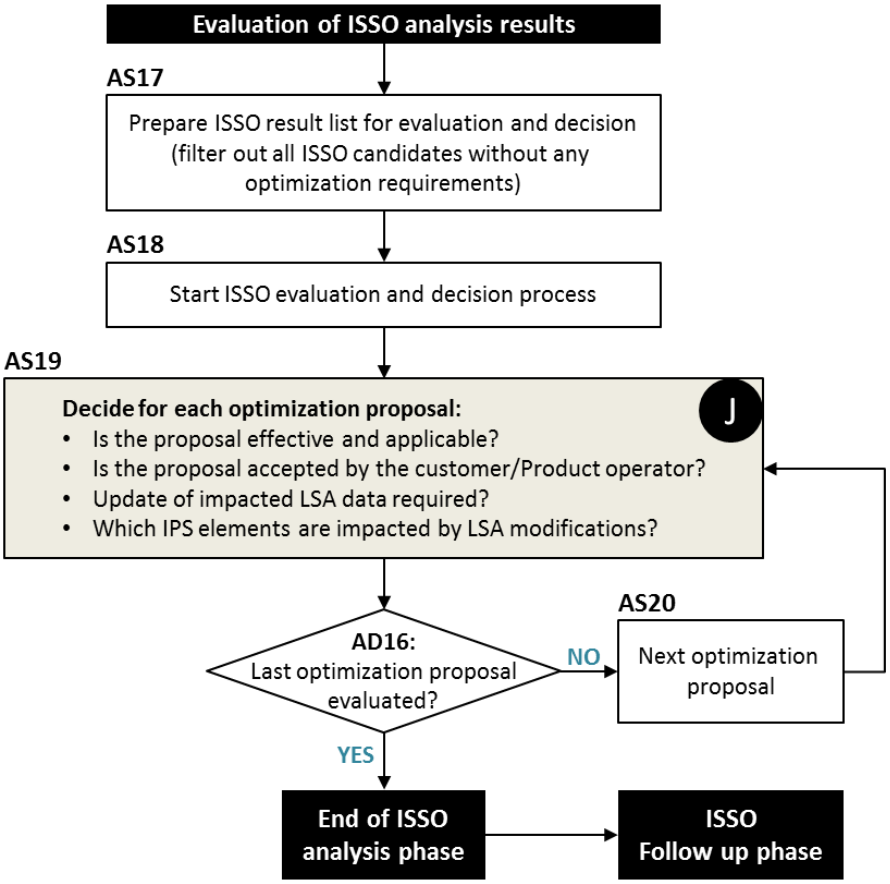
Fig 14 Detailed logic to analyze disposal tasks for explosive materials (class 1)

3.4.6 Evaluation of ISSO analysis results

After having analyzed all tasks from the ISSO candidate list, there will be a documented analysis result for each task. In principle, there are two general categories of analysis results:

- the task can remain unchanged, because the ISSO analysis (block **A** to **H**) did not generate any need or recommendations to modify the task or to perform other activities to improve task performance (eg, additional training of personnel, or update of the maintenance concept by changing the maintenance level at which the task is performed)
- the ISSO analysis (block **A** to **H**) generated needs or recommendations to modify the task or to perform other activities to optimize the task performance.

To complete the ISSO analysis phase, it is necessary to evaluate the task that have showed a potential for optimization. Refer to Fig 15.



ICN-B6865-S3000L0143-001-01

Fig 15 Finalization of ISSO analysis phase

Table 12 Process steps and decisions for finalization of ISSO analysis phase

Code	Description
AS17	The adequate preparation of the outcome of the ISSO analysis phase must consider all analyzed tasks that have showed a potential for optimization. It is necessary to divide the analyzed tasks in the Optimization Proposal List (OPL) in clusters (eg, Product systems), and sort them based on their significance with respect to Product use/maintenance. This ensures an effective evaluation and discussion of the issues together with the customer.

Code	Description
AS18	Start the evaluation of the ISSO analysis result with the first optimization proposal from the prepared list.
AS19	In the analysis block **J**, it is necessary to determine the decision logic at the basis of ISSO results evaluation. Before starting the ISSO process, the ISSO GD must include and document all the criteria for the evaluation of the optimization proposals. The logic can start, but is not limited to, the questions in Fig 15, which provide a good starting point. Using the determined logic, each optimization proposal will be "closed" by a final decision concerning the implementation (yes/no) and identification of the activities required to obtain the desired optimization.
AD16	If the last optimization proposal from the OPL has been evaluated and the actions to take have been decided, the ISSO analysis phase is finished.
AS20	Go to the next optimization proposal (end of OPL not reached yet)

At the end of the ISSO analysis phase, it is possible to plan the implementation of all actions resulting from the analysis performed in blocks **A** to **J**.

3.5 ISSO follow-up phase

3.5.1 Scope of the follow-up phase

In the follow-up phase, the actual implementation of modifications is one of the main activities. Additionally, it is possible to perform further analysis and/or activities to cover specific aspects. In this phase, it is possible to determine how to continue after performing all agreed actions.

3.5.2 Follow-up phase aspects

The follow-up phase can have different characteristics depending on the Product and the Product use scenario. For this reason, the ISSO GD must also plan and document the follow-up phase with all its activities.

The following list illustrates a series of aspects that can be part of the ISSO follow-up phase:

- Which activities must be planned and performed to monitor the success of the implemented changes within the tasks?
- Which activities must be planned and performed to monitor the success of additional training?
- Is there a significant change in the expected Product use scenario, which requires a reassessment of the support scenario (eg, maintenance scenario during a deployment or during operation under stressful conditions)?
- Were there any issues reported as in-service feedback from the customer which significantly exceeded the predicted/expected values of supportability KPI?
- Are there additional data/information reported by the customer/operator during the ISSO analysis phase that must be taken into consideration also for tasks already analyzed?
- Could it be beneficial to reassess some selected tasks for which no optimization potential was detected during the first ISSO analysis? This can have a significant impact, especially if the operators provided additional feedback during the ISSO analysis phase.
- Was there an unacceptable and excessive increase of cost caused by specific tasks/equipment/systems?
- Was there an unacceptable and excessive decrease of mission availability caused by specific tasks/equipment/systems?

For the severe problem initiators:

- Is it possible to improve the situation by including industry support in the corrective and/or preventive maintenance for a Product?

- Is it possible to improve the situation by introducing additional preventive maintenance tasks? Trigger PMA activities, for example based on the principles of S4000P.
- Is it possible to improve the situation by making changes to the design? Is a design change applicable, reasonable and affordable to overcome problem situations with Product operational support and/or Product maintenance?

4 Associated parts of the S3000L data model

The documentation of the associated data to this chapter is supported by the following Units of Functionality (UoF), refer to Chap 19:

- S3000L UoF Decision Tree Template Definition
- S3000L UoF In Service Optimization Analysis
- S3000L UoF Change Information

Chapter 18

Interrelations to other S-Series IPS specifications

Table of contents

List of tables

List of figures

References

Table 1 References

Chap No./Document No.	Title
Chap 1	Introduction to the specification
Chap 10	Development of a preventive maintenance program
Chap 17	In-Service LSA
Chap 20	Data exchange
S1000D	International specification for technical publications using a common source database
S1000X	Input data specification for S1000D
S2000M	International specification for material management
S2000X	Input data specification for S2000M
S3000X	Input data specification for S3000L
S4000P	International specification for developing and continuously improving preventive maintenance
S4000X	Input data specification for S4000P
S5000F	International specification for operational and maintenance data feedback
S6000T	International specification for training analysis and design
S6000X	Input data specification for S6000T
SX000i	International guide for the use of the S-Series Integrated Logistics Support (ILS) specifications
SX001G	Glossary for the S-Series IPS specifications
SX002D	Common data model for the S-Series IPS specifications
ASD-STE100	ASD Simplified Technical English

1 General

1.1 Introduction

The S3000L specification should not be considered a standalone entity. In the environment of the S-Series IPS specifications, there is a set of existing specifications. ASD-STE100 (Simplified Technical English) describes the coherent usage of English language within technical publication. S1000D describes the development and management of technical publication. S2000M describes the material management processes and procedures used for Product support. S4000P describes the analysis methodologies for Preventive Maintenance Analysis (PMA). ASD-STE100, S1000D and S2000M represent the first generation of the S-Series IPS specifications, S3000L and S4000P represent the second generation.

The third and latest generation of S-Series IPS specifications comprises S5000F, S6000T and SX000i. S5000F describes the exchange of in-service data and information, S6000T describes the training needs analysis and training design activities, and finally, SX000i, as an overarching specification, provides a guide how to use the S-Series IPS specifications in the context of an

IPS process. The S-Series IPS specifications covers the most important areas of Product supportability.

1.2 Purpose

This chapter provides an overview about how to harmonize the existing S-Series IPS specifications and S3000L. It illustrates the common ground and benefits of combining S3000L together with the S-Series IPS specifications S1000D, S2000M, S4000P, S5000F, S6000T and SX000i.

1.3 Scope

The target readers for this chapter are, for example, Integrated Product Support (IPS) managers, Logistics Support Analysis (LSA) managers and any other personnel involved in supportability analysis activities. This chapter also highlights the interdisciplinary character of product support development.

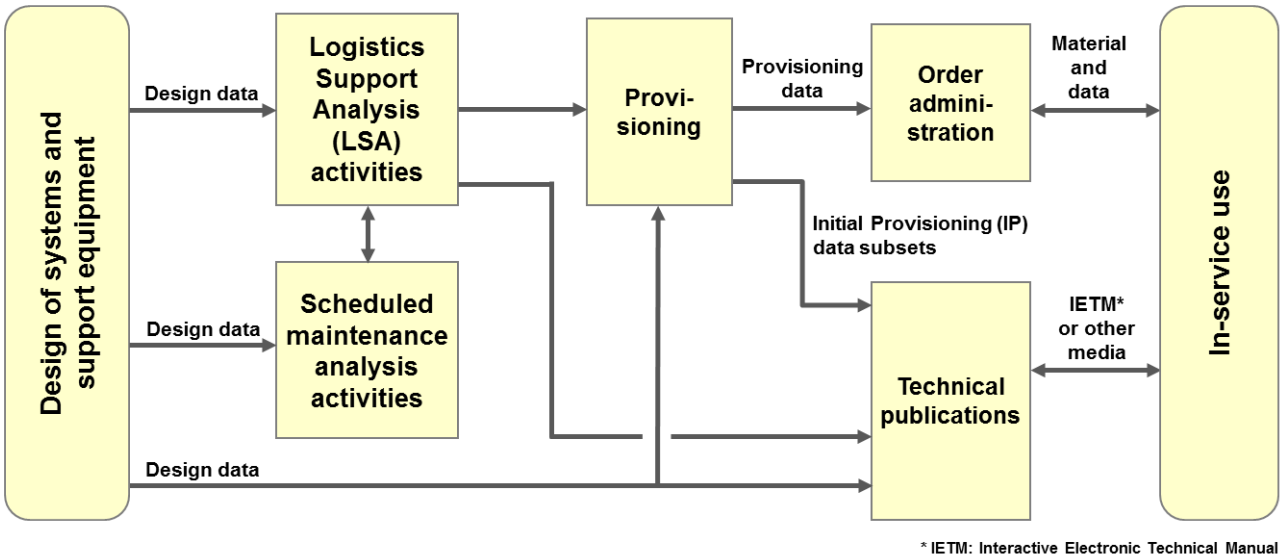
2 Benefits of using the S-Series IPS specifications

2.1 Conceptual background

The idea to develop a suite of ASD specifications to support said processes dates back to early 1993. High-level industrial and government stakeholders took part in an international NATO acquisition logistics workshop in Paris in 1993, defining and networking the major fields of Product support during an acquisition phase, usually for new products and major retrofit campaigns. The principle outcome of this workshop is illustrated in Fig 1. These fields provided the framework for the subsequent development of specifications, with the goal of covering the complete Product life cycle.

These major fields were:

- Logistics Support Analysis (LSA)
- scheduled maintenance analysis
- provisioning
- order administration
- technical publications



ICN-B6865-S3000L0064-004-01

Fig 1 Acquisition logistics main business processes (1)

At the time of the NATO workshop in Paris in 1993, S1000D already covered the technical publications domain, while S2000M covered the provisioning and order administration fields.

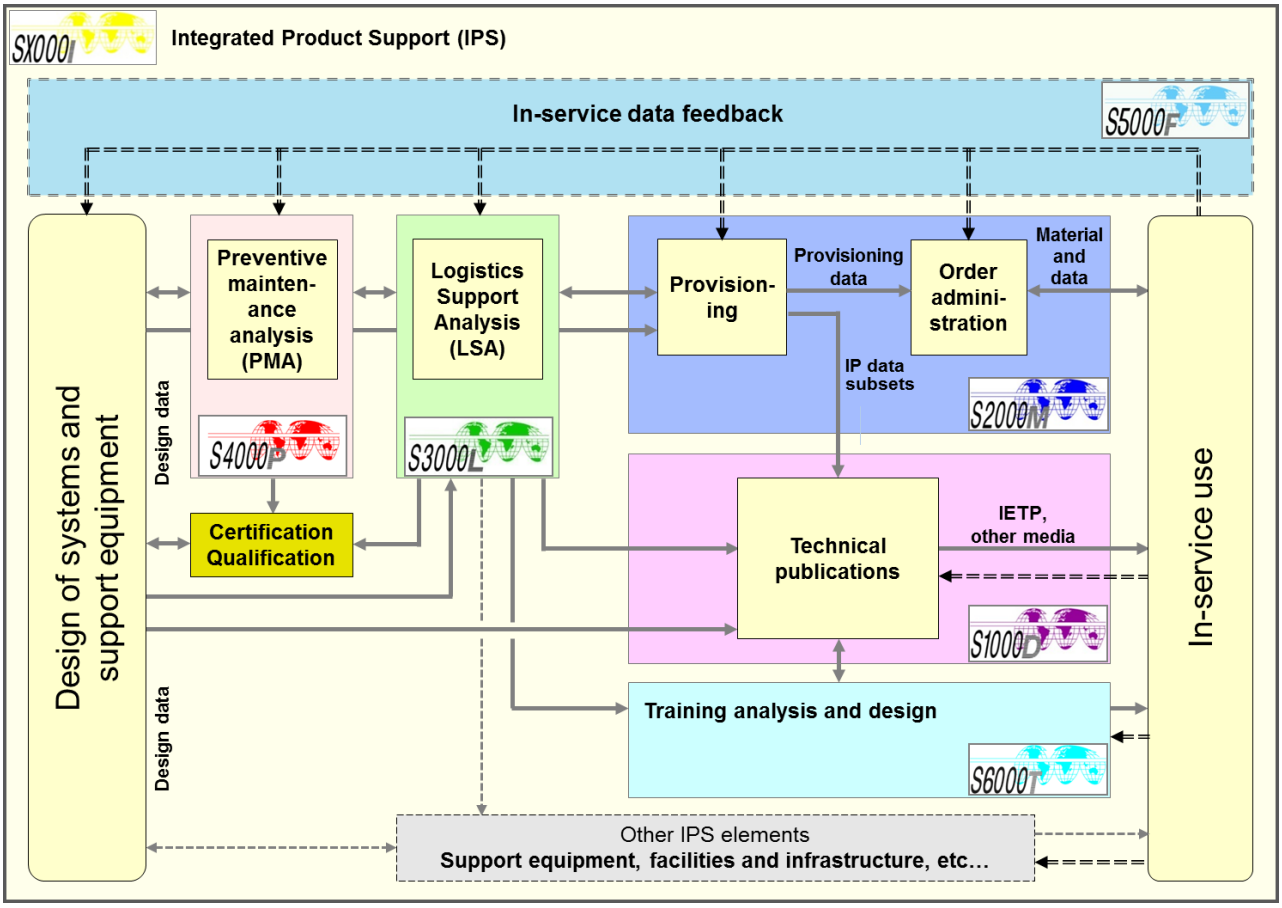
At that time no ASD specification existed for LSA and for scheduled maintenance analysis. In 2005, the ASD together with the Aerospace Industries Association of America (AIA) started the development of S3000L, a global specification for LSA. Refer to Chap 1.

It soon became clear that additional specifications would be necessary to satisfy the requirements for the support of the full Product life cycle. Those requirements are:

- analysis of preventive maintenance requirements (including the scheduled maintenance analysis field), covered by S4000P first published in May 2014
- in-service data/information feedback, covered by S5000F first published in September 2016
- identification of required training needs and develop training for Product operation and maintenance, covered by S6000T first published in February 2020

Additionally, the SX000i specification provides guidelines for the complete IPS process. Development started in September 2011 under the supervision of a team of operators, manufacturers and regulatory authorities. SX000i was first published in December 2015.

Today, the S-Series IPS specifications comprise seven main specifications covering the major support functions of the Product life cycle, as illustrated in Fig 2. This figure also relates back to Fig 1 to illustrate how the main fields identified in the past are covered today by the S-Series IPS specifications, including the fields which were added.



ICN-B6865-S3000L0065-003-01

Fig 2 Acquisition logistics main business processes (2)

Additional supplements that describe the data and information exchange processes (input data specifications, for example S3000X), an overall IPS glossary SX001G and a common data model SX002D (covering all data elements included in at least two specifications) continuously improve and complete the S-Series IPS specifications.

Note

Certification and qualification can also require inputs from IPS disciplines (eg, technical documentation, material support, support/test equipment).

Note

Fig 2 focuses on engineering support and IPS activities. However, in-service feedback can also be provided to other stakeholders (eg, customers or certification authorities).

2.2 Integrated concept of operation

The individual specifications of the S-Series IPS specifications are not stand-alone entities. In acquisition product support, the functional specifications serve as a coherent set of common data definitions and integrated data exchange. This integrated concept ensures all major functionalities are covered effectively and without any overlap. The specifications are internationally accepted and are broadly used for example in the aerospace and defense business, but are not limited to these areas.

S3000L plays a central role with respect to other specifications of the S-Series IPS specifications. It is the principal tool to:

- support the Product design relevant to reliability, maintainability, testability and support optimization of Life Cycle Costs (LCC)
- define all required resources to support the Product in its intended use and during the entire life cycle of the Product, focusing mainly on in-service operation
- maintain and update the basic Product support data and the Product maintenance concept during the entire service life

Because of the interfaces and overlap of data elements, it is highly recommended that all IPS guidance documents for all specifications are created in a harmonized way. This will prevent different definitions and interpretations, and will prevent missing or superfluous data.

3 Interrelation to engineering and supportability engineering

Each of the S-Series IPS specifications are closely linked to engineering, reliability, maintainability, testability, system safety and engineering change management. They do not cover these areas, which provide substantial input to the areas already covered by specifications. It is necessary to document both these interfaces and the processes that define baselines. A baseline is a configuration of a Product at a given time. This configuration is the basis for any IPS deliverable.

4 Interrelation to S1000D

4.1 Purpose of S1000D

S1000D satisfies the need for an international specification about the production and distribution of technical publication and learning content. This specification can cover the documentation of any type of military and civil Products.

Note

Since 2007, ASD, AIA and Air Transport Association of America (ATA) have jointly developed, maintained and promoted S1000D at a global level.

Note

The trade organization Airlines for America (A4A) took on the role of former Air Transport Association of America (ATA).

4.2 S3000L/S1000D

The LSA process develops information/data on corrective and preventive maintenance, as well as operational support tasks. Said information/data is the baseline for the data modules produced in accordance with S1000D. LSA data are also the input for the maintenance planning information.

The maintenance and operational tasks information/data must include, but is not limited to:

- task description
- preliminary requirements
  - required conditions
  - required resources (personnel, material, support equipment)

Maintenance publications use these data as a direct input for the relevant procedure. Depending on the publication production process, procedures can include some of these data directly.

If a full set of maintenance planning data is developed in the LSA process, it is possible to transfer these data to the maintenance planning data modules.

Note

Since it is possible to group tasks into one procedural data module or split them into several reusable procedural data modules, there is not always a one-to-one relation between a task originated in the LSA process and a procedural data module.

Issue 1.0 of S3000L, DEX1A&D, DEX3A&D and Issue 1.0 of the S1003X specification (S1000D, Issue 4.0 to S3000L, Issue 1.0 interchange specification) provided the original details of the data set to be exchanged. Chap 20 defines data exchange principles for Issue 2.0 of S3000L.

Note

S1003X was the first interchange specification between S1000D and S3000L and has now been replaced by S1000X which defines all inputs into S1000D from each of the S-Series IPS specifications. To continue this level of harmonization, a step-by-step re-organization of input specifications to develop data input requirements for each of the S-Series IPS specifications has been started. S3000X (based on the S3000L XML schema) will be the interface specification for S3000L and will describe the data/information requirements taken from other specifications (S1000D, S2000M, S4000P, S5000F and S6000T). Refer to Para 2.1.

Some of the other IPS Specifications will soon have their own input documents as well, which will include: S2000X, S4000X, S6000X.

5 Interrelation to S2000M

5.1 Purpose of S2000M

S2000M originally defined the material management processes and procedures for support activities for airborne and ground equipment used in the military field. The current issue of S2000M includes the business processes and data applicable to both military and commercial Products.

Widely used Enterprise Resource Planning (ERP) software (eg, SAP, Oracle) can cover the S2000M processes to a large extent. It is possible to use S2000M as an interchange specification to exchange material master data between two ERP systems.

The processes described within S2000M cover the interfaces between the contractor and the customers. Pursuant to the contractual terms, customers can be provided with the typical deliverables of material management, such as:

- provisioning
- NATO codification (as a special military requirement)
- procurement planning
- order administration
- invoicing
- repair administration

5.2 S3000L/S2000M

Information/data generated during S3000L LSA activities determine the range and depth of the Product corrective and preventive maintenance, as well as the required material resources during in-service operation. S2000M provisioning functionality best suits interfacing with S3000L. According to S2000M, the term “provisioning” indicates the process of selecting support items, spare parts and consumables necessary for all Product support activities.

The prime objective of that phase is provisioning data compilation for items identified as relevant for customer's maintenance and support activities.

Provisioning data are compiled according to the compilation rules specified in S2000M. Usually, these rules are confirmed or modified/specified during a relevant Guidance Conference (GC). The GC takes place at the start of any project that involves S2000M procedures and determines to what extent LSA data will be used to support the provisioning process.

All data that is developed via S3000L that is utilized by S2000M will be defined in the S2000X specification.

Note

It is recommended that data common to both S2000M and S3000L (eg, the name of a part/equipment) are harmonized. The IPS GC must determine the leading discipline for technical values (eg, a part identifier).

6 Interrelation to S4000P

6.1 Purpose of S4000P

S4000P provides methodologies to develop Preventive Maintenance Task Requirements (PMTR). These PMTR are the precondition to determine which preventive maintenance tasks will be applicable and effective for operators/manufacturers and will be accepted by regulatory authorities (if applicable). Experts from manufacturers and/or system integrating companies will jointly develop the PMTR and corresponding threshold details (eg, intervals, triggers).

Specifically, S4000P outlines the general organization and decision processes to define the initial PMTR for the Product life, as well as an ongoing improvement of preventive maintenance throughout the Product in-service phase, covered by In-Service Maintenance Optimization (ISMO).

6.2 S3000L/S4000P

Preventive maintenance focuses on the prevention of critical and catastrophic functional failures during operation and an adequate "reaction" on special events. S4000P provides structured and approved methodologies to define PMTR that allows operators to use a Product safely during its entire life cycle. It is important to mitigate identified issues as early as possible, with a view to limiting re-design efforts and designing an applicable and effective preventive maintenance program at acceptable costs. PMTR are the outcome of the S4000P process and are further detailed during task identification and Maintenance Task Analysis (MTA) within the LSA process.

It is possible to associate an interval with a PMTR, which is named Preventive Maintenance Task Requirement Interval (PMTRI). Chap 10 describes the packaging process to adapt PMTRI. A PMTR associated with a special event is named Preventive Maintenance Task Requirement Event (PMTRE). LSA considers all these PMTR as corresponding task requirements that are

associated with the corresponding rectifying preventive maintenance tasks. This workflow from S4000P to S3000L constitutes a strong relationship between S3000L and S4000P and will be defined within the upcoming S3000X input specification.

During the in-service phase, it is necessary to continuously scrutinize the efficiency of the current preventive maintenance program, and optimize/adapt as required. For this purpose, it is recommended that the ISMO process from S4000P be used.

A close configuration management of Product breakdown and part information between the specifications is required to support the Product safely and effectively. Refer to Chap 10.

7 Interrelation to S5000F

7.1 Purpose of S5000F

S5000F provides the in-service feedback to all relevant stakeholders with respect to in-service data and information collected and prepared by operators/customers, with a view to supporting and improving Product maintenance and operations (refer to Fig 2).

7.2 S3000L/S5000F

It is necessary that the in-service data/information (resulting from an S5000F process) and maintenance concept/tasks and Product support requirements (resulting from an S3000L LSA process) be compared on a regular basis to identify any need for re-evaluation or adaptation of current support solutions. Such a comparison clarifies whether the predictions of supportability analysis processes reflect the daily experience of the Product operators. It is necessary to collect and compare in-service data and the existing maintenance solution, in order to optimize the support solutions. The In-Service Support Optimization (ISSO) process in S3000L is the recommended process to structure the optimization, refer to Chap 17.

The required information for optimization differs on a case-by-case basis. S5000F has defined a set of common use cases that require the exchange of information. The definition of a data model supports these use cases in exchanging data/information in a standardized way. S5000F provides the unfiltered data/information without processing or condensing it. Feedback data/information evaluation is part of the requiring processes, for S3000L for example the ISSO process.

S3000X will identify the use cases and data required as inputs from S5000F to support S3000L requirements.

8 Interrelation to S6000T

8.1 Purpose of S6000T

S6000T defines all levels of analysis and training design to deliver relevant and effective Product training. This includes the Training Situation Analysis (TSA) and Training Needs Analysis (TNA), respectively.

8.2 S3000L/S6000T

LSA data provide an overview of all maintenance and support tasks, and how they must be performed, as a result of the MTA. This LSA data is a key input to perform TSA and TNA, respectively. In particular, the TNA requires information to decide whether a task needs specific training and if so, how the training can be provided. Each support task identified and analyzed during the LSA process is checked concerning training needs.

Typically, LSA data already contain information about required competence of the maintainer/operator performing the task. Additionally, LSA data contain details how to perform the task (including warnings and cautions) and which material resources are required (eg, support equipment, spare parts, consumables).

S3000L LSA data is the main information source to support the initial analysis activities in the frame of S6000T and will be documented as S6000T inputs via S6000X.

9 Interrelation to SX000i

9.1 Purpose of SX000i

SX000i aims to establish a common understanding of IPS and defines a common and generic IPS process throughout the life cycle of a Product.

SX000i provides an explanation how to apply the S-Series IPS specifications in an IPS program. It provides a set of suitable IPS activities to enable a standard-based communication and information/data exchange among involved stakeholders to achieve the main IPS objectives:

- design the Product for optimal support
- initiate the most cost-effective support solution
- acquire and provide the support for the Product

An LSA program based on S3000L is a part of an IPS program and is a binding data/information source for processes defined by the other S-Series IPS specifications.

9.2 S3000L/SX000i

LSA and IPS are closely related. As described in this specification, the LSA process is the central tool to achieve an integrated development of deliverables of the IPS elements in close relation to the design of the supported Product. SX000i describes the connection between LSA, as a central supportability analysis process, and IPS as an effective management process over the entire lifetime of complex and durable Products.

Chapter 19

Data model

Table of contents

List of tables

List of figures

References

Table 1 References

Chap No./Document No.	Title
Chap 3	LSA process
Chap 4	Product structures and change management in LSA
Chap 7	Corrective maintenance analysis
Chap 8	Damage and special event analysis
Chap 9	Operational support analysis
Chap 10	Development of a preventive maintenance program
Chap 11	Level of repair analysis
Chap 12	Task requirements and maintenance task analysis
Chap 15	Obsolescence analysis
Chap 16	Disposal
Chap 17	In-service LSA
Chap 20	Data exchange
Chap 22	Data element list
S4000P	International specification for developing and continuously improving preventive maintenance
SX002D	Common data model for the S-Series IPS specifications
SX004G	Unified Modeling Language (UML) model reader's guide
SX005G	S-Series IPS specifications XML schema implementation guidance
www.uml.org	Unified Modeling Language

1 General

1.1 Introduction

This chapter defines a coherent data model for the data that can be exchanged between the Logistics Support Analysis (LSA) process as defined in S3000L and related business processes (refer to Chap 20).

The S3000L data model is described using the UML 2.0™ (Unified Modeling Language) class model diagrams (www.uml.org) and is based on SX002D Common Data Model (CDM) issue 2.1. Conventions used for the data model are described in SX004G, including descriptions of the data types that are used for the respective class attributes.

Each class and attribute in the S3000L data model is defined in Chap 22.

1.2 Scope

The following areas are in scope of the data model:

- Definition of the LSA program and the Products to be supported (refer to Chap 3)
- Document the breakdown structure for the Product and its Product variants (refer to Chap 4)
- Support the early phases of the LSA program in terms of selecting the LSA candidate items and identification of analysis activities to be performed for each LSA candidate (refer to Chap 3)
- Support the identification of task requirements resulting from eg, corrective maintenance analysis (refer to Chap 7), damage and special event analysis (refer to Chap 8) and operational support analysis (refer to Chap 9)
- Document the preventive maintenance program and the packaging of preventive maintenance tasks (refer to Chap 10)
- Document the outcome from the Level of Repair Analysis (LORA) (refer to Chap 11)
- Document the results from the task requirements and Maintenance Task Analysis (MTA) (refer to Chap 12)
- Document the results from obsolescence and disposal analysis (refer to Chap 15 and Chap 16)
- Support the In-Service Support Optimization (ISSO) process (refer to Chap 17)

1.3 Out of scope

The data model has been defined to support one project (LSA program) at a time. Any implementations of the data model in a software application that will support multiple projects need to define how data can be reused between projects as part of the software development.

The data model does not provide full support for all the identified candidate item analysis activities, but focuses on Product breakdown definition (refer to Chap 4), corrective maintenance analysis (refer to Chap 7), damage and special event analysis (refer to Chap 8), task requirements and maintenance task analysis (refer to Chap 12) and in-service LSA (refer to Chap 17). Results from other analysis activities, eg, LORA, can be referenced by summarized data, descriptions and/or by inclusion of digital document files or document references.

The data model does not cover all data elements needed to support the LSA process, but just those data elements that will typically be exchanged between a contractor and subcontractors, partners and customers.

2 Data model overview

The S3000L data model is organized into a set of Unit of Functionalities (UoF) which divides the overall data model for S3000L into a set of smaller data models. Each UoF defines classes, relationships and attributes required to document a specific aspect of LSA. The purpose of dividing the data model into a set of UoFs is to present small and coherent portions of the data model, and to gradually give the reader an understanding of the complete data model.

The order in which the respective UoF is presented in Para 3 does not follow the chapters of S3000L, but is organized in an alphabetical order. However, there is a suggested order of reading that support the LSA process as laid out in Chap 3 and the rest of this specification. The suggested reading order is:

- Overall product, project, operational and maintenance context data
  - UoF Product and Project, refer to Para 3.27
  - UoF Product Usage Context, refer to Para 3.29
  - UoF Facility, refer to Para 3.13
  - UoF Location, refer to Para 3.19
  - UoF Environment Definition, refer to Para 3.12
- Product structure and part characteristics
  - UoF Breakdown Structure, refer to Para 3.3
  - UoF Part Definition, refer to Para 3.25
  - UoF Aggregated Element, refer to Para 3.1
  - UoF Hardware Element, refer to Para 3.17
  - UoF Software Element, refer to Para 3.34
  - UoF Zone Element, refer to Para 3.41
  - UoF Product Design Configuration, refer to Para 3.28
- LSA candidate selection, requirements and analysis activities
  - UoF Performance Parameter, refer to Para 3.26
  - UoF LSA Candidate, refer to Para 3.21
- Corrective maintenance analysis
  - UoF Failure Mode, refer to Para 3.14
  - UoF LSA Failure Mode Group, refer to Para 3.22
  - UoF Failure Mode Symptom, refer to Para 3.16
  - UoF Failure Mode Isolation, refer to Para 3.15
- Damage and special event analysis
  - UoF Product Usage Phase, refer to Para 3.30
  - UoF Special Event, refer to Para 3.35
  - UoF Damage Definition, refer to Para 3.7
- Task requirements and influence on design
  - UoF Task Requirement, Refer to Para 3.37
  - UoF Time Limit, Refer to Para 3.40
  - UoF Design Change Request, Refer to Para 3.9
- Maintenance task analysis
  - UoF Task, refer to Para 3.36
  - UoF Circuit Breaker, refer to Para 3.5
  - UoF Task Resource, refer to Para 3.38
  - UoF Resource Specification, refer to Para 3.32
  - UoF Competence Definition, refer to Para 3.6
  - UoF Task Usage, refer to Para 3.39
- In-service LSA
  - UoF Decision Tree Template Definition, refer to Para 3.8

- UoF In Service Optimization Analysis, refer to Para 3.18

– General capabilities

- UoF Change Information, refer to Para 3.4
- UoF Security Classification, refer to Para 3.33
- UoF Organization Assignment, refer to Para 3.24
- UoF Document, refer to Para 3.11
- UoF Digital File, refer to Para 3.10
- UoF Remark, refer to Para 3.31
- UoF Applicability Statement, refer to Para 3.2

– LSA messages

- UoF Message, refer to Para 3.23
- UoF Logistics Support Analysis Message Content, refer to Para 3.20

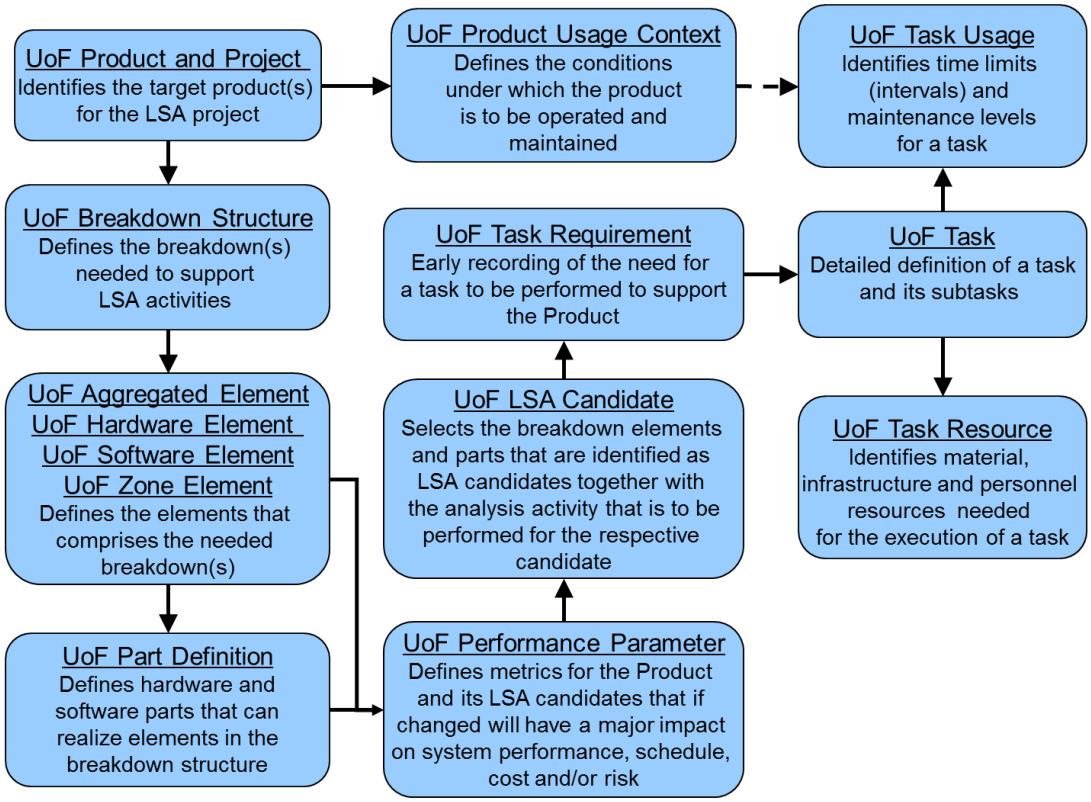
Each UoF contains:

– An overall description of the UoF
– A graphical representation of the class model for the UoF
– Definitions of classes and interfaces identified within the UoF

A complete listing shows that the S3000L data model contains 41 UoFs. Each UoF adds to the capability to capture and share data relevant to the LSA process.

The suggested reading order follows the process where the first data elements defined are those required to define the project and the Product in focus for the LSA activities. This is then followed by recording data that is defined during the respective LSA analysis activity leading up to a set of defined task requirements. The end result of S3000L is a set of detailed task definitions including subtasks, resources, time limits and information about where the task is to be performed.

Fig 1 provides an overview of the main UoFs, and how they are interrelated:



ICN-B6865-S3000L0214-004-01

Fig 1 S3000L UoF overview

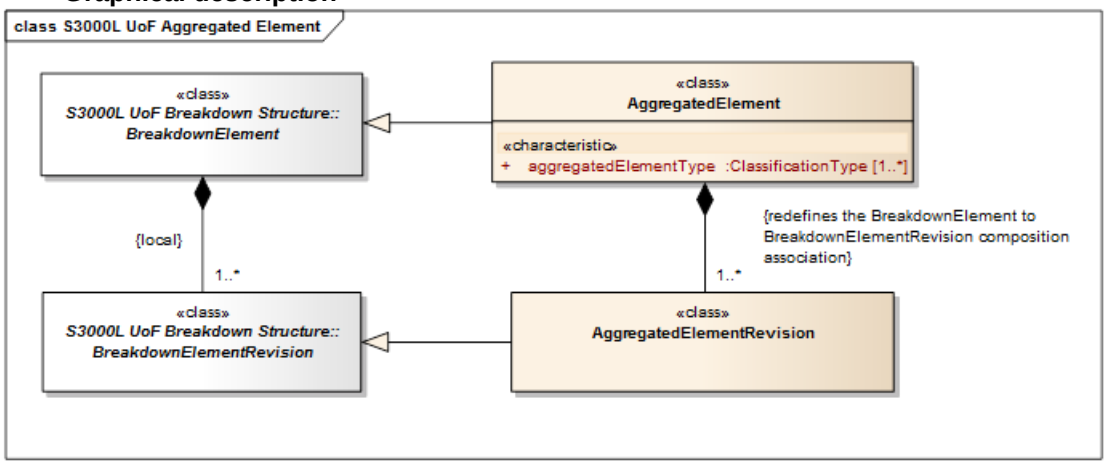
3 Data model units of functionality

3.1 S3000L UoF Aggregated Element

3.1.1 Description

The Aggregated Element UoF provides the capability to specify that an element within a breakdown represents a collection of elements for an identified purpose.

3.1.2 Graphical description



ICN-B6865-S3000L0221-003-01

Fig 2 S3000L UoF Aggregated Element

3.1.3 Class definition

3.1.3.1 AggregatedElement

AggregatedElement is a BreakdownElement (refer to Para 3.3) that is a container for a collection of BreakdownElements which are grouped for an identified purpose.

3.1.3.1.1 Attribute(s)

This class has the following attributes:

- breakdownElementIdentifier (inherited from BreakdownElement), one or many
- breakdownElementName (inherited from BreakdownElement), zero, one or many
- breakdownElementEssentiality (inherited from BreakdownElement), zero or one
- aggregatedElementType, one or many

3.1.3.1.2 Associations

This class has the following associations:

- An aggregate association with one or many instances of AggregatedElementRevision

3.1.3.1.3 Implementations

This class implements the following <<extend>> interfaces:

- AnalysisCandidateItem (inherited from BreakdownElement). Refer to Para 3.21
- BreakdownElementInZoneItem (inherited from BreakdownElement). Refer to Para 3.41
- DecisionTreeAnalysisItem (inherited from BreakdownElement). Refer to Para 3.8
- DetectionMeansAlarmItem (inherited from BreakdownElement). Refer to Para 3.16
- DigitalFileReferencingItem (inherited from BreakdownElement). Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- FailureModeAnalysisItem (inherited from BreakdownElement). Refer to Para 3.14
- InServiceOptimizationAnalysisItem (inherited from BreakdownElement). Refer to Para 3.18
- MeasurementPointDefinitionItem (inherited from BreakdownElement). Refer to Para 3.16
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- PerformanceParameterItem (inherited from BreakdownElement). Refer to Para 3.26
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem (inherited from BreakdownElement). Refer to Para 3.33

3.1.3.2 AggregatedElementRevision

AggregatedElementRevision is a BreakdownElementRevision (refer to Para 3.3) representing an iteration applied to an AggregatedElement.

3.1.3.2.1 Attribute(s)

This class has the following attributes:

- breakdownElementRevisionIdentifier (inherited from BreakdownElementRevision)
- breakdownElementDescription (inherited from BreakdownElementRevision), zero, one or many
- maintenanceSignificantOrRelevant (inherited from BreakdownElementRevision)
- breakdownElementRevisionRationale (inherited from BreakdownElementRevision), zero, one or many
- breakdownElementRevisionDate (inherited from BreakdownElementRevision), zero or one
- breakdownElementRevisionStatus (inherited from BreakdownElementRevision), zero or one

3.1.3.2.2 Associations

This class has the following associations:

- A directed has association with zero, one or many instances of BreakdownElementRevisionRelationship (inherited from BreakdownElementRevision). Refer to Para 3.3.

3.1.3.2.3 Implementations

This class implements the following <<extend>> interfaces:

- AnalysisCandidateItem (inherited from BreakdownElementRevision). Refer to Para 3.21
- BreakdownElementInZoneItem (inherited from BreakdownElementRevision). Refer to Para 3.41
- ChangeControlledItem (inherited from BreakdownElementRevision). Refer to Para 3.4
- DetectionMeansAlarmItem (inherited from BreakdownElementRevision). Refer to Para 3.16
- DigitalFileReferencingItem (inherited from BreakdownElementRevision). Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- FailureModeAnalysisItem (inherited from BreakdownElementRevision). Refer to Para 3.14
- InServiceOptimizationAnalysisItem (inherited from BreakdownElementRevision). Refer to Para 3.18
- MeasurementPointDefinitionItem (inherited from BreakdownElementRevision). Refer to Para 3.16
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- PerformanceParameterItem (inherited from BreakdownElementRevision). Refer to Para 3.26

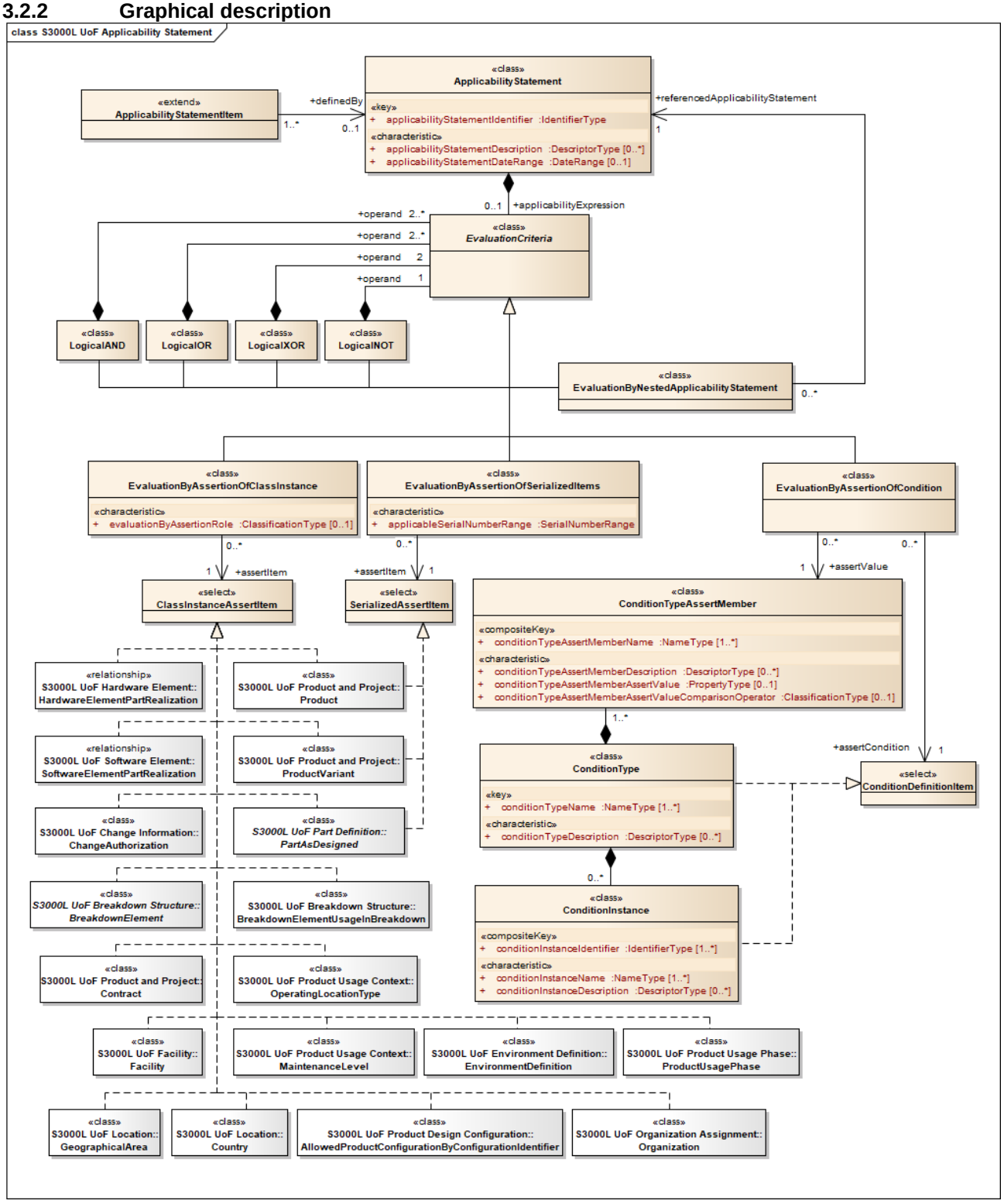
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- TaskAnalysisItem (inherited from BreakdownElementRevision). Refer to Para 3.39
- TaskRequirementAnalysisItem (inherited from BreakdownElementRevision). Refer to Para 3.37

3.2 S3000L UoF Applicability Statement

3.2.1 Description

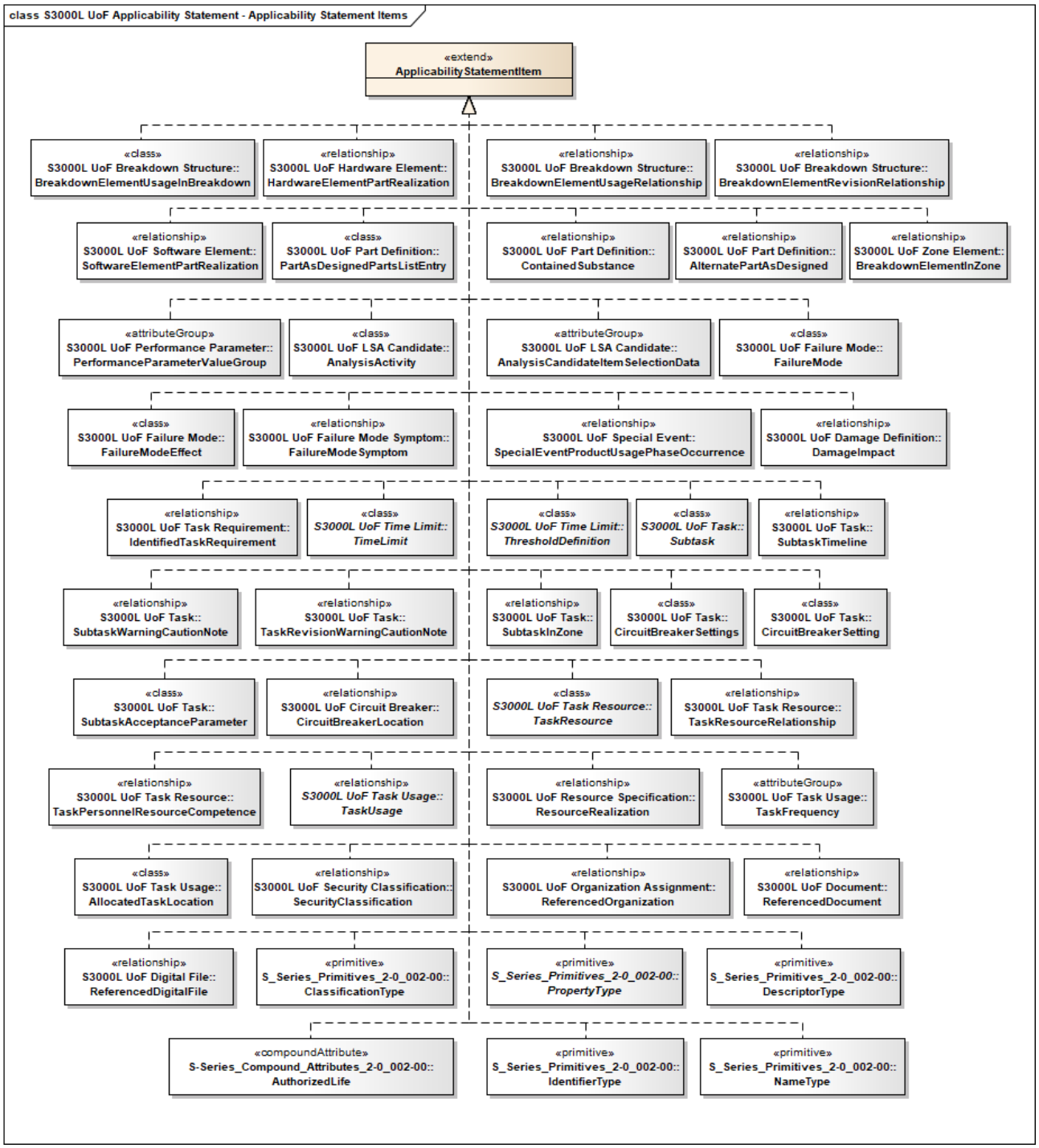
The Applicability Statement UoF provides the capability to define the situation or situations under which related items are valid.

3.2.2 Graphical description



ICN-B6865-S3000L0235-004-01

Fig 3 S3000L UoF Applicability Statement



ICN-B6865-S3000L0266-002-01

Fig 4 S3000L UoF Applicability Statement - Applicability Statement Items

3.2.3 Class definition

3.2.3.1 ApplicabilityStatement

`ApplicabilityStatement` is a `<<class>>` that defines the situation or situations under which related items are valid.

3.2.3.1.1 Attribute(s)

This class has the following attributes:

- applicabilityStatementIdentifier
- applicabilityStatementDescription, zero, one or many
- applicabilityStatementDateRange, zero or one

3.2.3.1.2 Associations

This class has the following associations:

- An aggregate association with zero or one instance of EvaluationCriteria

3.2.3.1.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.2.3.2 ApplicabilityStatementItem

ApplicabilityStatementItem is an <<extend>> interface that provides its associated data model to those classes which can have restricted validity as defined by an associated ApplicabilityStatement.

3.2.3.2.1 Class members

Classes that implement the ApplicabilityStatementItem <<extend>> interface are:

- AllocatedTaskLocation. Refer to Para 3.39
- AlternatePartAsDesigned. Refer to Para 3.25
- AnalysisActivity. Refer to Para 3.21
- AnalysisCandidateItemSelectionData. Refer to Para 3.21
- AuthorizedLife. Refer to SX004G
- BreakdownElementInZone. Refer to Para 3.41
- BreakdownElementRevisionRelationship. Refer to Para 3.3
- BreakdownElementUsageInBreakdown. Refer to Para 3.3
- BreakdownElementUsageRelationship. Refer to Para 3.3
- CircuitBreakerLocation. Refer to Para 3.5
- CircuitBreakerSetting. Refer to Para 3.36
- CircuitBreakerSettings. Refer to Para 3.36
- ClassificationType. Refer to SX004G
- ContainedSubstance. Refer to Para 3.25
- DamageImpact. Refer to Para 3.7
- DescriptorType. Refer to SX004G
- FailureMode. Refer to Para 3.14
- FailureModeEffect. Refer to Para 3.14
- FailureModeSymptom. Refer to Para 3.16
- HardwareElementPartRealization. Refer to Para 3.17
- IdentifiedTaskRequirement. Refer to Para 3.37
- IdentifierType. Refer to SX004G
- NameType. Refer to SX004G
- PartAsDesignedPartsListEntry. Refer to Para 3.25

- PerformanceParameterValueGroup. Refer to Para 3.26
- PropertyType. Refer to SX004G
- ReferencedDigitalFile. Refer to Para 3.10
- ReferencedDocument. Refer to Para 3.11
- ReferencedOrganization. Refer to Para 3.24
- ResourceRealization. Refer to Para 3.32
- SecurityClassification. Refer to Para 3.33
- SoftwareElementPartRealization. Refer to Para 3.34
- SpecialEventProductUsagePhaseOccurrence. Refer to Para 3.35
- Subtask. Refer to Para 3.36
- SubtaskAcceptanceParameter. Refer to Para 3.36
- SubtaskInZone. Refer to Para 3.36
- SubtaskTimeline. Refer to Para 3.36
- SubtaskWarningCautionNote. Refer to Para 3.36
- TaskFrequency. Refer to Para 3.39
- TaskPersonnelResourceCompetence. Refer to Para 3.38
- TaskResource. Refer to Para 3.38
- TaskResourceRelationship. Refer to Para 3.38
- TaskRevisionWarningCautionNote. Refer to Para 3.36
- TaskUsage. Refer to Para 3.39
- ThresholdDefinition. Refer to Para 3.40
- TimeLimit. Refer to Para 3.40

3.2.3.2.2 Associations

The ApplicabilityStatementItem <<extend>> interface has the following associations:

- A directed definedBy association with zero or one instance of ApplicabilityStatement

3.2.3.3 ClassInstanceAssertItem

ClassInstanceAssertItem is a <<select>> interface that identifies classes from which an instance can be used as the EvaluationByAssertionOfClassInstance assert item.

3.2.3.3.1 Class members

This <<select>> interface includes the following class members:

- AllowedProductConfigurationByConfigurationIdentifier. Refer to Para 3.28
- BreakdownElement. Refer to Para 3.3
- BreakdownElementUsageInBreakdown. Refer to Para 3.3
- ChangeAuthorization. Refer to Para 3.4
- Contract. Refer to Para 3.27
- Country. Refer to Para 3.19
- EnvironmentDefinition. Refer to Para 3.12
- Facility. Refer to Para 3.13
- GeographicalArea. Refer to Para 3.19
- HardwareElementPartRealization. Refer to Para 3.17
- MaintenanceLevel. Refer to Para 3.29
- OperatingLocationType. Refer to Para 3.29

- Organization. Refer to Para 3.24
- PartAsDesigned. Refer to Para 3.25
- Product. Refer to Para 3.27
- ProductUsagePhase. Refer to Para 3.30
- ProductVariant. Refer to Para 3.27
- SoftwareElementPartRealization. Refer to Para 3.34

3.2.3.4 ConditionDefinitionItem

ConditionDefinitionItem is a <<select>> interface that identifies classes from which an instance can be used as the EvaluationByAssertionOfCondition assert condition.

3.2.3.4.1 Class members

This <<select>> interface includes the following class members:

- ConditionInstance
- ConditionType

3.2.3.5 ConditionInstance

ConditionInstance is a <<class>> that defines an individual concept or object having the characteristics of a generic ConditionType.

Example(s)

- Uniquely identified service bulletin

3.2.3.5.1 Attribute(s)

This class has the following attributes:

- conditionInstanceIdentifier, one or many
- conditionInstanceName, one or many
- conditionInstanceDescription, zero, one or many

3.2.3.5.2 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.2.3.6 ConditionType

ConditionType is a <<class>> that defines a concept or an object that needs to be included in applicability statements where the concept or object is not already represented in the data model.

Example(s)

- Environmental conditions

3.2.3.6.1 Attribute(s)

This class has the following attributes:

- conditionTypeName, one or many
- conditionTypeDescription, zero, one or many

3.2.3.6.2 *Associations*
This class has the following associations:

- An aggregate association with zero, one or many instances of ConditionInstance
- An aggregate association with one or many instances of ConditionTypeAssertMember

3.2.3.6.3 *Implementations*
This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.2.3.7 ConditionTypeAssertMember
ConditionTypeAssertMember is <<class>> that defines a member for a given ConditionType which can be mapped to a Boolean expression and be evaluated to be either TRUE or FALSE.

3.2.3.7.1 *Attribute(s)*
This class has the following attributes:

- conditionTypeAssertMemberName, one or many
- conditionTypeAssertMemberDescription, zero, one or many
- conditionTypeAssertMemberAssertValue, zero or one
- conditionTypeAssertMemberAssertValueComparisonOperator, zero or one

3.2.3.7.2 *Implementations*
This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.2.3.8 EvaluationByAssertionOfClassInstance
EvaluationByAssertionOfClassInstance is an EvaluationCriteria that identifies a class instance to be used as an assert item and be mapped to a Boolean expression which can be evaluated to be either TRUE or FALSE.

3.2.3.8.1 *Attribute(s)*
This class has the following attributes:

- evaluationByAssertionRole, zero or one

3.2.3.8.2 *Associations*
This class has the following associations:

- A directed assertItem association with one instance of ClassInstanceAssertItem

3.2.3.8.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.2.3.9 EvaluationByAssertionOfCondition

EvaluationByAssertionOfCondition is an EvaluationCriteria that identifies a combination of a defined condition and a defined value to be used as an assert item and be mapped to a Boolean expression which can be evaluated to be either TRUE or FALSE.

3.2.3.9.1 Associations

This class has the following associations:

- A directed assertValue association with one instance of ConditionTypeAssertMember
- A directed assertCondition association with one instance of ConditionDefinitionItem

3.2.3.9.2 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.2.3.10 EvaluationByAssertionOfSerializedItems

EvaluationByAssertionOfSerializedItems is an EvaluationCriteria that identifies a class instance together with an associated serial number range to be used as an assert item and be mapped to a Boolean expression which can be evaluated to be either TRUE or FALSE.

3.2.3.10.1 Attribute(s)

This class has the following attributes:

- applicableSerialNumberRange

3.2.3.10.2 Associations

This class has the following associations:

- A directed assertItem association with one instance of SerializedAssertItem

3.2.3.10.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24

- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.2.3.11 EvaluationByNestedApplicabilityStatement

EvaluationByNestedApplicabilityStatement is an EvaluationCriteria that enables an ApplicabilityStatement to be reused as part of this EvaluationCriteria.

Note

This class enables the definition of nested applicability statements.

3.2.3.11.1 Associations

This class has the following associations:

- A directed referencedApplicabilityStatement association with one instance of ApplicabilityStatement

3.2.3.11.2 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.2.3.12 EvaluationCriteria

EvaluationCriteria is a <<class>> that defines conditions that can be mapped to a Boolean expression which can be evaluated to be either TRUE or FALSE.

3.2.3.12.1 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.2.3.13 LogicalAND

LogicalAND is an EvaluationCriteria that defines a Boolean operation where the results of all its associated EvaluationCriteria must be TRUE for the result to be TRUE, otherwise the result is FALSE.

3.2.3.13.1 Associations

This class has the following associations:

- An aggregate operand association with two or more instances of EvaluationCriteria

3.2.3.13.2 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.2.3.14 LogicalNOT

LogicalNOT is an EvaluationCriteria that defines a Boolean operation where the result from its associated EvaluationCriteria must be FALSE for the result to be TRUE, otherwise the result is FALSE.

3.2.3.14.1 Associations

This class has the following associations:

- An aggregate operand association with one instance of EvaluationCriteria

3.2.3.14.2 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.2.3.15 LogicalOR

LogicalOR is an EvaluationCriteria that defines a Boolean operation where the result from at least one of its associated EvaluationCriteria must be TRUE for the result to be TRUE, otherwise the result is FALSE.

3.2.3.15.1 Associations

This class has the following associations:

- An aggregate operand association with two or more instances of EvaluationCriteria

3.2.3.15.2 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.2.3.16 LogicalXOR

LogicalXOR is an EvaluationCriteria that defines a Boolean operation where the result from one and only one of its associated EvaluationCriteria must be TRUE for the result to be TRUE, otherwise the result is FALSE.

3.2.3.16.1 Associations

This class has the following associations:

- An aggregate operand association with exactly two instances of EvaluationCriteria

3.2.3.16.2 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.2.3.17 SerializedAssertItem

SerializedAssertItem is a <<select>> interface that identifies classes from which an instance can be used as the EvaluationByAssertionOfSerializedItems assert item.

3.2.3.17.1 Class members

This <<select>> interface includes the following class members:

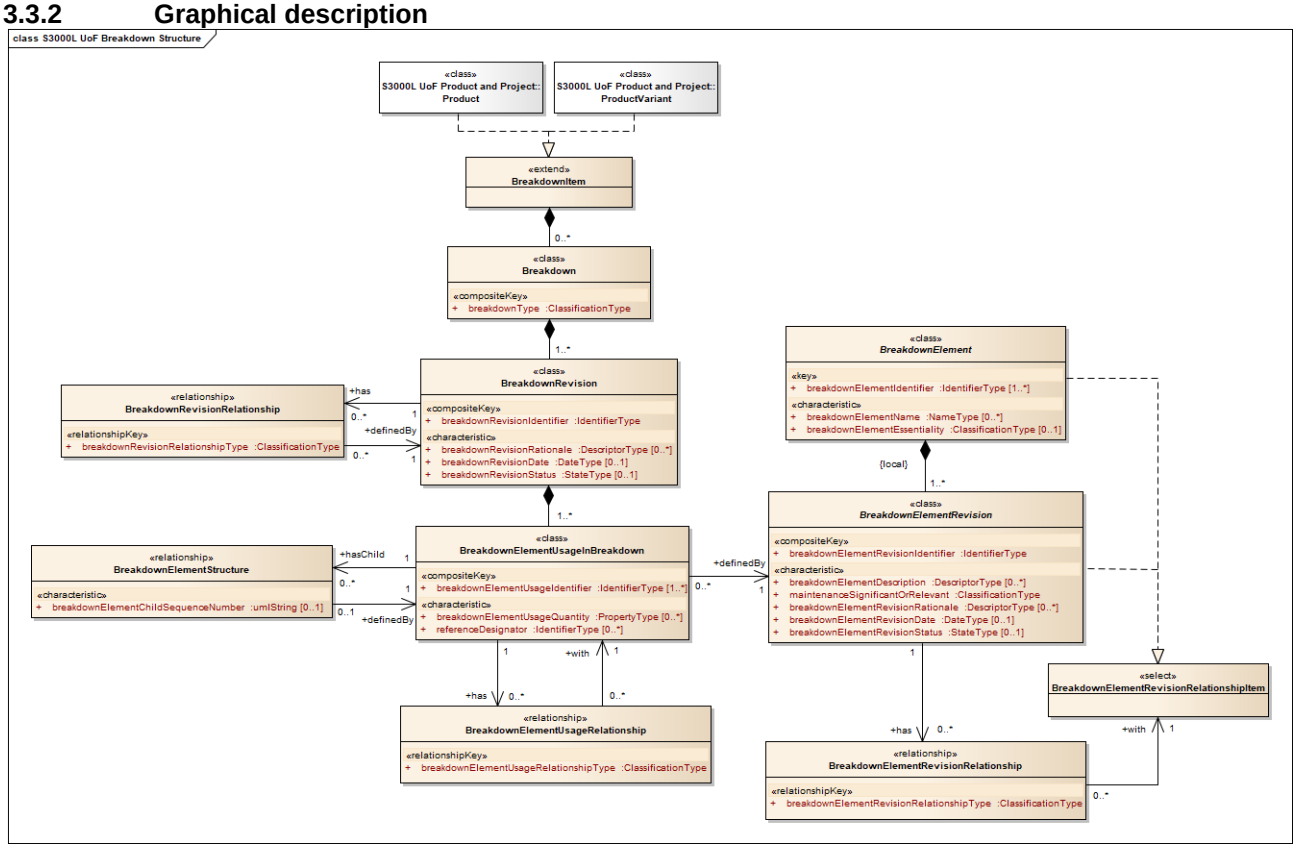
- PartAsDesigned. Refer to Para 3.25
- Product. Refer to Para 3.27
- ProductVariant. Refer to Para 3.27

3.3 S3000L UoF Breakdown Structure

3.3.1 Description

The Breakdown Structure UoF provides the capability to define any number of hierarchical structures for a specific Product or Product variant.

3.3.2 Graphical description



ICN-B6865-S3000L0217-004-01

Fig 5 S3000L UoF Breakdown Structure

3.3.3 Class definition

3.3.3.1 Breakdown

`Breakdown` is a `<<class>>` that identifies a specific partitioning of a `Product` (refer to Para 3.27) to form a parent-child structure of related instances of `BreakdownElement`.

3.3.3.1.1 Attribute(s)

This class has the following attributes:

- `breakdownType`

3.3.3.1.2 Associations

This class has the following associations:

- An aggregate association with one or many instances of `BreakdownRevision`

3.3.3.1.3 Implementations

This class implements the following `<<extend>>` interfaces:

- `DigitalFileReferencingItem`. Refer to Para 3.10
- `DocumentReferencingItem` (inherited from `BaseObject`). Refer to Para 3.11
- `OrganizationReferencingItem` (inherited from `BaseObject`). Refer to Para 3.24
- `ProjectSpecificExtensionItem` (inherited from `BaseObject`). Refer to SX002D
- `RemarkItem` (inherited from `BaseObject`). Refer to Para 3.31
- `SecurityClassificationItem`. Refer to Para 3.33

3.3.3.2 BreakdownElement

BreakdownElement is a <<class>> defining a partition of a Product that is used in one or many instances of Breakdown.

3.3.3.2.1 Attribute(s)

This class has the following attributes:

- breakdownElementIdentifier, one or many
- breakdownElementName, zero, one or many
- breakdownElementEssentiality, zero or one

3.3.3.2.2 Associations

This class has the following associations:

- An aggregate association with one or many instances of BreakdownElementRevision

3.3.3.2.3 Implementations

This class implements the following <<extend>> interfaces:

- AnalysisCandidateItem. Refer to Para 3.21
- BreakdownElementInZoneItem. Refer to Para 3.41
- DecisionTreeAnalysisItem. Refer to Para 3.8
- DetectionMeansAlarmItem. Refer to Para 3.16
- DigitalFileReferencingItem. Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- FailureModeAnalysisItem. Refer to Para 3.14
- InServiceOptimizationAnalysisItem. Refer to Para 3.18
- MeasurementPointDefinitionItem. Refer to Para 3.16
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- PerformanceParameterItem. Refer to Para 3.26
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem. Refer to Para 3.33

3.3.3.3 BreakdownElementRevision

BreakdownElementRevision is a <<class>> representing an iteration applied to a BreakdownElement.

3.3.3.3.1 Attribute(s)

This class has the following attributes:

- breakdownElementRevisionIdentifier
- breakdownElementDescription, zero, one or many
- maintenanceSignificantOrRelevant
- breakdownElementRevisionRationale, zero, one or many
- breakdownElementRevisionDate, zero or one
- breakdownElementRevisionStatus, zero or one

3.3.3.3.2 Associations

This class has the following associations:

- A directed has association with zero, one or many instances of BreakdownElementRevisionRelationship

3.3.3.3.3 Implementations

This class implements the following <<extend>> interfaces:

- AnalysisCandidateItem. Refer to Para 3.21
- BreakdownElementInZoneItem. Refer to Para 3.41
- ChangeControlledItem. Refer to Para 3.4
- DetectionMeansAlarmItem. Refer to Para 3.16
- DigitalFileReferencingItem. Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- FailureModeAnalysisItem. Refer to Para 3.14
- InServiceOptimizationAnalysisItem. Refer to Para 3.18
- MeasurementPointDefinitionItem. Refer to Para 3.16
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- PerformanceParameterItem. Refer to Para 3.26
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- TaskAnalysisItem. Refer to Para 3.39
- TaskRequirementAnalysisItem. Refer to Para 3.37

3.3.3.4 BreakdownElementRevisionRelationship

BreakdownElementRevisionRelationship is a <<relationship>> where one BreakdownElementRevision relates to another BreakdownElement or BreakdownElementRevision.

3.3.3.4.1 Attribute(s)

This class has the following attributes:

- breakdownElementRevisionRelationshipType

3.3.3.4.2 Associations

This class has the following associations:

- A directed with association with one instance of BreakdownElementRevisionRelationshipItem

3.3.3.4.3 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.3.3.5 BreakdownElementRevisionRelationshipItem

BreakdownElementRevisionRelationshipItem is a <<select>> interface that provides the capability to be associated with a BreakdownElementRevision.

3.3.3.5.1 Class members

This <<select>> interface includes the following class members:

- BreakdownElement
- BreakdownElementRevision

3.3.3.6 BreakdownElementStructure

BreakdownElementStructure is a <<relationship>> that establishes a hierarchical structure between two usages of BreakdownElement that belong to the same BreakdownRevision.

3.3.3.6.1 Attribute(s)

This class has the following attributes:

- breakdownElementChildSequenceNumber, zero or one

3.3.3.6.2 Associations

This class has the following associations:

- A directed definedBy association with one instance of BreakdownElementUsageInBreakdown

3.3.3.6.3 Implementations

This class implements the following <<extend>> interfaces:

- ChangeControlledItem. Refer to Para 3.4
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.3.3.7 BreakdownElementUsageInBreakdown

BreakdownElementUsageInBreakdown is a <<class>> that represents a member of a BreakdownRevision.

Note

A BreakdownElementRevision can belong to multiple BreakdownRevisions.

3.3.3.7.1 Attribute(s)

This class has the following attributes:

- breakdownElementUsageIdentifier, one or many
- breakdownElementUsageQuantity, zero, one or many
- referenceDesignator, zero, one or many

3.3.3.7.2 Associations

This class has the following associations:

- A directed definedBy association with one instance of BreakdownElementRevision
- A directed hasChild association with zero, one or many instances of BreakdownElementStructure

- A directed has association with zero, one or many instances of BreakdownElementUsageRelationship

3.3.3.7.3 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- BreakdownElementInZoneItem. Refer to Para 3.41
- ChangeControlledItem. Refer to Para 3.4
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- EffectiveOnProductConfigurationItem. Refer to Para 3.28
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- UsableOnItem. Refer to Para 3.28

3.3.3.8 BreakdownElementUsageRelationship

BreakdownElementUsageRelationship is a <<relationship>> where one usage of a BreakdownElement relates to the usage of another BreakdownElement.

Note

Both related instances of BreakdownElementUsageInBreakdown must reside within the same BreakdownRevision.

Example(s)

- Version C of a radio is restricted to the use of software version B in breakdown revision 2

3.3.3.8.1 Attribute(s)

This class has the following attributes:

- breakdownElementUsageRelationshipType

3.3.3.8.2 Associations

This class has the following associations:

- A directed with association with one instance of BreakdownElementUsageInBreakdown

3.3.3.8.3 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.3.3.9 BreakdownItem

BreakdownItem is an <<extend>> interface that provides its associated data model to those classes that implement it.

3.3.3.9.1 *Class members*
Classes that implement the BreakdownItem <<extend>> interface are:

- Product. Refer to Para 3.27
- ProductVariant. Refer to Para 3.27

3.3.3.9.2 *Associations*
The BreakdownItem <<extend>> interface has the following associations:

- An aggregate association with zero, one or many instances of Breakdown

3.3.3.10 BreakdownRevision
BreakdownRevision is a <<class>> representing an iteration applied to a Breakdown.

Note
BreakdownRevision is used to document design iterations and not breakdown variants.

3.3.3.10.1 *Attribute(s)*
This class has the following attributes:

- breakdownRevisionIdentifier
- breakdownRevisionRationale, zero, one or many
- breakdownRevisionDate, zero or one
- breakdownRevisionStatus, zero or one

3.3.3.10.2 *Associations*
This class has the following associations:

- An aggregate association with one or many instances of BreakdownElementUsageInBreakdown
- A directed has association with zero, one or many instances of BreakdownRevisionRelationship

3.3.3.10.3 *Implementations*
This class implements the following <<extend>> interfaces:

- ChangeControlledItem. Refer to Para 3.4
- DigitalFileReferencingItem. Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.3.3.11 BreakdownRevisionRelationship
BreakdownRevisionRelationship is a <<relationship>> where one BreakdownRevision relates to another BreakdownRevision.

3.3.3.11.1 *Attribute(s)*
This class has the following attributes:

- breakdownRevisionRelationshipType

3.3.3.11.2 Associations

This class has the following associations:

- A directed `definedBy` association with one instance of `BreakdownRevision`

3.3.3.11.3 Implementations

This class implements the following `<<extend>>` interfaces:

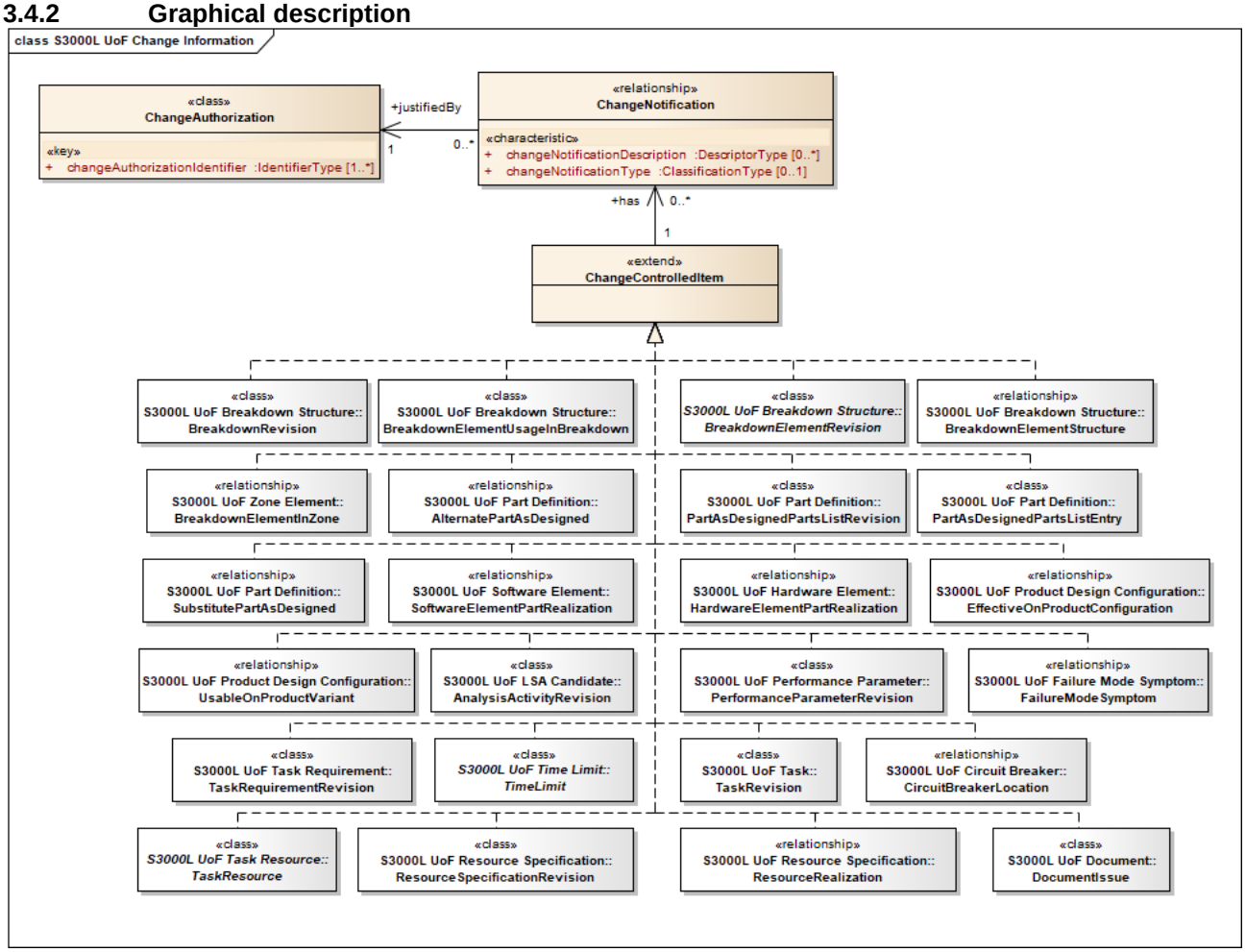
- `DocumentReferencingItem` (inherited from `BaseObject`). Refer to Para 3.11
- `OrganizationReferencingItem` (inherited from `BaseObject`). Refer to Para 3.24
- `ProjectSpecificExtensionItem` (inherited from `BaseObject`). Refer to SX002D
- `RemarkItem` (inherited from `BaseObject`). Refer to Para 3.31

3.4 S3000L UoF Change Information

3.4.1 Description

The Change Information UoF provides the capability to identify that an item has been affected by a change authorization.

3.4.2 Graphical description



ICN-B6865-S3000L0291-001-01

Fig 6 S3000L UoF Change Information

3.4.3 Class definition

3.4.3.1 ChangeAuthorization

ChangeAuthorization is a <<class>> that is the record of the permission to modify product design, its procedures and/or associated product support data.

3.4.3.1.1 Attribute(s)

This class has the following attributes:

- changeAuthorizationIdentifier, one or many

3.4.3.1.2 Implementations

This class implements the following <<extend>> interfaces:

- DigitalFileReferencingItem. Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.4.3.2 ChangeControlledItem

ChangeControlledItem is an <<extend>> interface that provides its associated data model to those classes that can be affected by a ChangeAuthorization.

3.4.3.2.1 Class members

Classes that implement the ChangeControlledItem <<extend>> interface are:

- AlternatePartAsDesigned. Refer to Para 3.25
- AnalysisActivityRevision. Refer to Para 3.21
- BreakdownElementInZone. Refer to Para 3.41
- BreakdownElementRevision. Refer to Para 3.3
- BreakdownElementStructure. Refer to Para 3.3
- BreakdownElementUsageInBreakdown. Refer to Para 3.3
- BreakdownRevision. Refer to Para 3.3
- CircuitBreakerLocation. Refer to Para 3.5
- DocumentIssue. Refer to Para 3.11
- EffectiveOnProductConfiguration. Refer to Para 3.28
- FailureModeSymptom. Refer to Para 3.16
- HardwareElementPartRealization. Refer to Para 3.17
- PartAsDesignedPartsListEntry. Refer to Para 3.25
- PartAsDesignedPartsListRevision. Refer to Para 3.25
- PerformanceParameterRevision. Refer to Para 3.26
- ResourceRealization. Refer to Para 3.32
- ResourceSpecificationRevision. Refer to Para 3.32
- SoftwareElementPartRealization. Refer to Para 3.34
- SubstitutePartAsDesigned. Refer to Para 3.25
- TaskRequirementRevision. Refer to Para 3.37
- TaskResource. Refer to Para 3.38
- TaskRevision. Refer to Para 3.36
- TimeLimit. Refer to Para 3.40
- UsableOnProductVariant. Refer to Para 3.28

3.4.3.2.2 Associations

The ChangeControlledItem <<extend>> interface has the following associations:

- A directed has association with zero, one or many instances of ChangeNotification

3.4.3.3 ChangeNotification

ChangeNotification is a <<relationship>> that identifies an item changed due to the associated ChangeAuthorization.

3.4.3.3.1 Attribute(s)

This class has the following attributes:

- changeNotificationDescription, zero, one or many
- changeNotificationType, zero or one

3.4.3.3.2 Associations

This class has the following associations:

- A directed justifiedBy association with one instance of ChangeAuthorization

3.4.3.3.3 Implementations

This class implements the following <<extend>> interfaces:

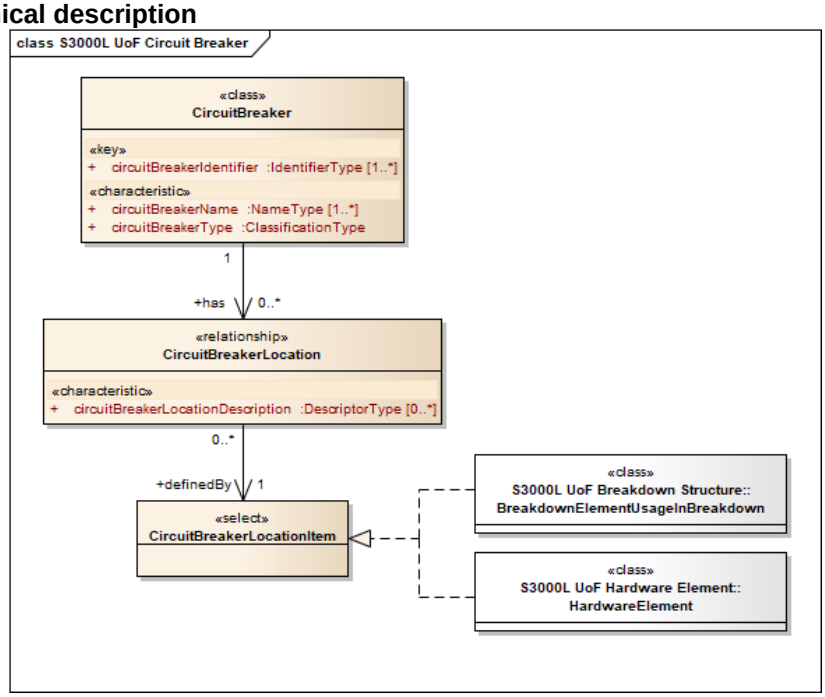
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.5 S3000L UoF Circuit Breaker

3.5.1 Description

The Circuit Breaker UoF provides the capability to define circuit breakers and their locations.

3.5.2 Graphical description



ICN-B6865-S3000L0284-001-01

Fig 7 S3000L UoF Circuit Breaker

3.5.3 Class definition

3.5.3.1 CircuitBreaker

`CircuitBreaker` is a `<<class>>` that represents an individual circuit breaker identified in the context of a defined `Product` (refer to Para 3.27).

3.5.3.1.1 Attribute(s)

This class has the following attributes:

- `circuitBreakerIdentifier`, one or many
- `circuitBreakerName`, one or many
- `circuitBreakerType`

3.5.3.1.2 Associations

This class has the following associations:

- A directed `has` association with zero, one or many instances of `CircuitBreakerLocation`

3.5.3.1.3 Implementations

This class implements the following `<<extend>>` interfaces:

- `DigitalFileReferencingItem`. Refer to Para 3.10
- `DocumentReferencingItem` (inherited from `BaseObject`). Refer to Para 3.11
- `OrganizationReferencingItem` (inherited from `BaseObject`). Refer to Para 3.24
- `ProjectSpecificExtensionItem` (inherited from `BaseObject`). Refer to SX002D
- `RemarkItem` (inherited from `BaseObject`). Refer to Para 3.31

3.5.3.2 CircuitBreakerLocation

CircuitBreakerLocation is a <<relationship>> that identifies the item on which the CircuitBreaker is placed.

3.5.3.2.1 Attribute(s)

This class has the following attributes:

- circuitBreakerLocationDescription, zero, one or many

3.5.3.2.2 Associations

This class has the following associations:

- A directed definedBy association with one instance of CircuitBreakerLocationItem

3.5.3.2.3 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- ChangeControlledItem. Refer to Para 3.4
- DigitalFileReferencingItem. Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.5.3.3 CircuitBreakerLocationItem

CircuitBreakerLocationItem is a <<select>> interface that identifies items which can be selected in order to determine the location for a CircuitBreaker.

3.5.3.3.1 Class members

This <<select>> interface includes the following class members:

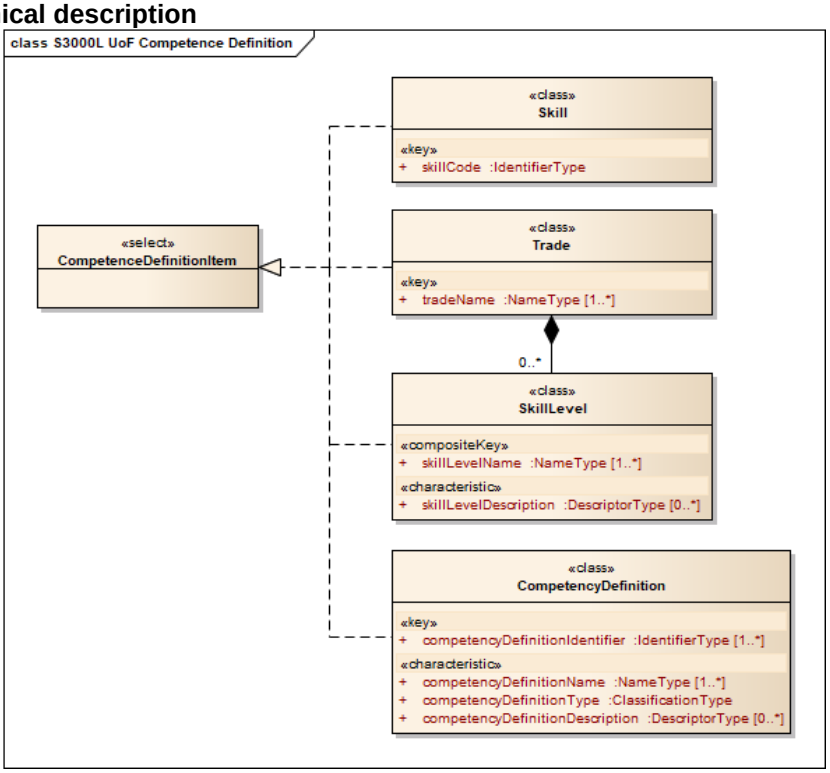
- BreakdownElementUsageInBreakdown. Refer to Para 3.3
- HardwareElement. Refer to Para 3.17

3.6 S3000L UoF Competence Definition

3.6.1 Description

The Competence Definition UoF supports the definition of abilities, craft, profession, and proficiency.

3.6.2 Graphical description



ICN-B6865-S3000L0286-001-01

Fig 8 S3000L UoF Competence Definition

3.6.3 Class definition

3.6.3.1 CompetenceDefinitionItem

`CompetenceDefinitionItem` is a `<<select>>` interface that identifies items which define measurable or observable possession of knowledge and skills.

3.6.3.1.1 Class members

This `<<select>>` interface includes the following class members:

- `CompetencyDefinition`
- `Skill`
- `SkillLevel`
- `Trade`

3.6.3.2 CompetencyDefinition

`CompetencyDefinition` is a `<<class>>` that represents the specification of measurable or observable knowledge, skills, and/or abilities necessary for successful performance by a person in a given context.

Note

Competency definition is broadly defined to include assertions of academic, professional, occupational, vocational and life goals, outcomes, and standards, however labeled

3.6.3.2.1 Attribute(s)

This class has the following attributes:

- `competencyDefinitionIdentifier`, one or many
- `competencyDefinitionName`, one or many
- `competencyDefinitionType`

- competencyDefinitionDescription, zero, one or many

3.6.3.2.2 Implementations

This class implements the following <<extend>> interfaces:

- DigitalFileReferencingItem. Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem. Refer to Para 3.33

3.6.3.3 Skill

Skill is a <<class>> that represents human cognitive, psychomotor, and affective abilities.

3.6.3.3.1 Attribute(s)

This class has the following attributes:

- skillCode

3.6.3.3.2 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem. Refer to Para 3.33

3.6.3.4 SkillLevel

SkillLevel is a <<class>> that represents a defined proficiency of a Trade.

3.6.3.4.1 Attribute(s)

This class has the following attributes:

- skillLevelName, one or many
- skillLevelDescription, zero, one or many

3.6.3.4.2 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.6.3.5 Trade

Trade is a <<class>> that represents a craft or profession which requires specific skills.

3.6.3.5.1 *Attribute(s)*
This class has the following attributes:

- `tradeName`, one or many

3.6.3.5.2 *Associations*
This class has the following associations:

- An aggregate association with zero, one or many instances of `SkillLevel`

3.6.3.5.3 *Implementations*
This class implements the following `<<extend>>` interfaces:

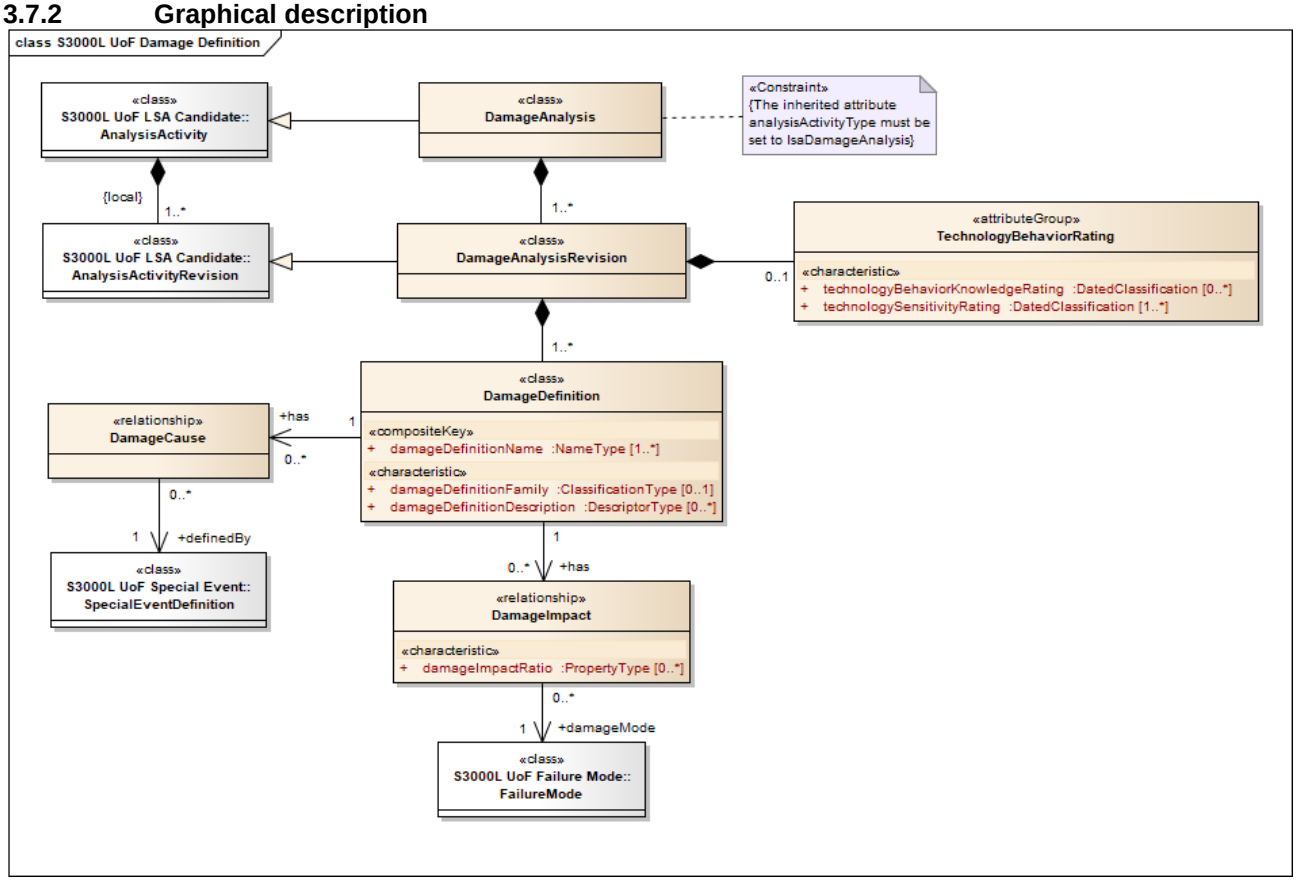
- `DocumentReferencingItem` (inherited from `BaseObject`). Refer to Para 3.11
- `OrganizationReferencingItem` (inherited from `BaseObject`). Refer to Para 3.24
- `ProjectSpecificExtensionItem` (inherited from `BaseObject`). Refer to SX002D
- `RemarkItem` (inherited from `BaseObject`). Refer to Para 3.31
- `SecurityClassificationItem`. Refer to Para 3.33

3.7 S3000L UoF Damage Definition

3.7.1 Description

The Damage Definition UoF provides the capability to define damages that can be induced on a Product during its in-service phase.

3.7.2 Graphical description



ICN-B6865-S3000L0282-001-01

Fig 9 S3000L UoF Damage Definition

3.7.3 Class definition

3.7.3.1 DamageAnalysis

DamageAnalysis is an AnalysisActivity (refer to Para 3.21) that represents the objective for, and outcome of, a damage analysis carried out for the AnalysisCandidateItem (refer to Para 3.21).

3.7.3.1.1 Attribute(s)

This class has the following attributes:

- analysisActivityType (inherited from AnalysisActivity)

3.7.3.1.2 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of DamageAnalysisRevision

3.7.3.1.3 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem (inherited from AnalysisActivity). Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem (inherited from AnalysisActivity). Refer to Para 3.33

3.7.3.2 DamageAnalysisRevision

DamageAnalysisRevision is an AnalysisActivityRevision (refer to Para 3.21) representing an iteration applied to a DamageAnalysis.

3.7.3.2.1 Attribute(s)

This class has the following attributes:

- analysisActivityRevisionIdentifier (inherited from AnalysisActivityRevision)
- analysisActivityDecision (inherited from AnalysisActivityRevision)
- analysisActivityDecisionRationale (inherited from AnalysisActivityRevision), zero, one or many
- analysisActivityStatusDescription (inherited from AnalysisActivityRevision), zero, one or many
- analysisActivityRevisionRationale (inherited from AnalysisActivityRevision), zero, one or many
- analysisActivityRevisionDate (inherited from AnalysisActivityRevision), zero or one
- analysisActivityRevisionStatus (inherited from AnalysisActivityRevision), zero or one

3.7.3.2.2 Associations

This class has the following associations:

- An aggregate association with one or many instances of DamageDefinition
- An aggregate association with zero, one or many instances of the TechnologyBehaviorRating <<attributeGroup>>

3.7.3.2.3 Implementations

This class implements the following <<extend>> interfaces:

- ChangeControlledItem (inherited from AnalysisActivityRevision). Refer to Para 3.4
- DigitalFileReferencingItem (inherited from AnalysisActivityRevision). Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.7.3.3 DamageCause

DamageCause is a <<relationship>> where a DamageDefinition relates to a SpecialEventDefinition (refer to Para 3.35) that in some way is associated with the damage.

3.7.3.3.1 Associations

This class has the following associations:

- A directed definedBy association with one instance of SpecialEventDefinition. Refer to Para 3.35

3.7.3.3.2 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.7.3.4 DamageDefinition

DamageDefinition is a <<class>> that represents a loss or reduction of functionality due to external causes or use outside specified limits.

3.7.3.4.1 Attribute(s)

This class has the following attributes:

- damageDefinitionName, one or many
- damageDefinitionFamily, zero or one
- damageDefinitionDescription, zero, one or many

3.7.3.4.2 Associations

This class has the following associations:

- A directed has association with zero, one or many instances of DamageCause
- A directed has association with zero, one or many instances of DamageImpact

3.7.3.4.3 Implementations

This class implements the following <<extend>> interfaces:

- DigitalFileReferencingItem. Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.7.3.5 DamageImpact

DamageImpact is a <<relationship>> that defines a consequence resulting from the defined DamageDefinition.

3.7.3.5.1 Attribute(s)

This class has the following attributes:

- damageImpactRatio, zero, one or many

3.7.3.5.2 Associations

This class has the following associations:

- A directed damageMode association with one instance of FailureMode. Refer to Para 3.14

3.7.3.5.3 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.7.3.6 TechnologyBehaviorRating

TechnologyBehaviorRating is an <<attributeGroup>> that collects attributes which categorizes the AnalysisCandidateItem (refer to Para 3.21) based on knowledge on technology sensitivity and behavior perspective.

3.7.3.6.1 Attribute(s)

This <<attributeGroup>> has the following attributes:

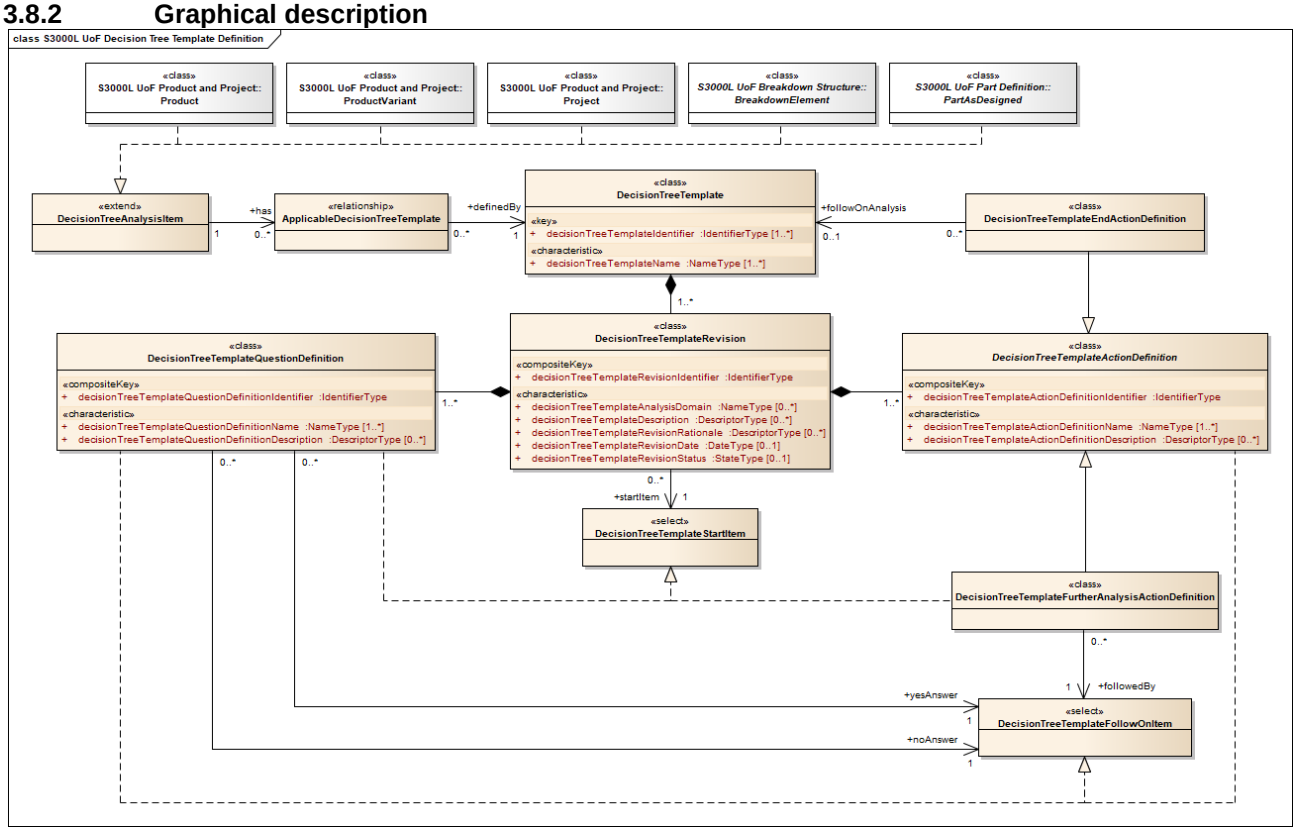
- technologyBehaviorKnowledgeRating, zero, one or many
- technologySensitivityRating, one or many

3.8 S3000L UoF Decision Tree Template Definition

3.8.1 Description

The Decision Tree Template Definition UoF provides the capability to represent the definition of decision processes which must be followed when performing an actual analysis.

3.8.2 Graphical description



ICN-B6865-S3000L0289-001-01

Fig 10 S3000L UoF Decision Tree Template Definition

3.8.3 Class definition

3.8.3.1 ApplicableDecisionTreeTemplate

ApplicableDecisionTreeTemplate is a <<relationship>> that identifies a DecisionTreeTemplate which can be used for the DecisionTreeAnalysisItem.

3.8.3.1.1 Associations

This class has the following associations:

- A directed definedBy association with one instance of DecisionTreeTemplate

3.8.3.1.2 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.8.3.2 DecisionTreeAnalysisItem

DecisionTreeAnalysisItem is an <<extend>> interface that provides its associated data model to those classes that implement it.

3.8.3.2.1 *Class members*
Classes that implement the DecisionTreeAnalysisItem <<extend>> interface are:

- BreakdownElement. Refer to Para 3.3
- PartAsDesigned. Refer to Para 3.25
- Product. Refer to Para 3.27
- ProductVariant. Refer to Para 3.27
- Project. Refer to Para 3.27

3.8.3.2.2 *Associations*
The DecisionTreeAnalysisItem <<extend>> interface has the following associations:

- A directed has association with zero, one or many instances of ApplicableDecisionTreeTemplate

3.8.3.3 DecisionTreeTemplate
DecisionTreeTemplate is a <<class>> that enables the representation of a decision process including a set of defined steps and binary decisions.

3.8.3.3.1 *Attribute(s)*
This class has the following attributes:

- decisionTreeTemplateIdentifier, one or many
- decisionTreeTemplateName, one or many

3.8.3.3.2 *Associations*
This class has the following associations:

- An aggregate association with one or many instances of DecisionTreeTemplateRevision

3.8.3.3.3 *Implementations*
This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.8.3.4 DecisionTreeTemplateActionDefinition
DecisionTreeTemplateActionDefinition is a <<class>> that specifies actions and/or measures that must be taken as part of the decision process.

3.8.3.4.1 *Attribute(s)*
This class has the following attributes:

- decisionTreeTemplateActionDefinitionIdentifier
- decisionTreeTemplateActionDefinitionName, one or many
- decisionTreeTemplateActionDefinitionDescription, zero, one or many

3.8.3.4.2 *Implementations*
This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.8.3.5 DecisionTreeTemplateEndActionDefinition

DecisionTreeTemplateEndActionDefinition is a DecisionTreeTemplateActionDefinition that defines an end action for the decision process.

3.8.3.5.1 Attribute(s)

This class has the following attributes:

- decisionTreeTemplateActionDefinitionIdentifier (inherited from DecisionTreeTemplateActionDefinition)
- decisionTreeTemplateActionDefinitionName (inherited from DecisionTreeTemplateActionDefinition), one or many
- decisionTreeTemplateActionDefinitionDescription (inherited from DecisionTreeTemplateActionDefinition), zero, one or many

3.8.3.5.2 Associations

This class has the following associations:

- A directed followOnAnalysis association with zero or one instance of DecisionTreeTemplate

3.8.3.5.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.8.3.6 DecisionTreeTemplateFollowOnItem

DecisionTreeTemplateFollowOnItem is a <<select>> interface that identifies items which can be selected to define the next step in the decision process.

3.8.3.6.1 Class members

This <<select>> interface includes the following class members:

- DecisionTreeTemplateActionDefinition
- DecisionTreeTemplateQuestionDefinition

3.8.3.7 DecisionTreeTemplateFurtherAnalysisActionDefinition

DecisionTreeTemplateFurtherAnalysisActionDefinition is a DecisionTreeTemplateActionDefinition that will be followed by another question or action.

3.8.3.7.1 Attribute(s)

This class has the following attributes:

- decisionTreeTemplateActionDefinitionIdentifier (inherited from DecisionTreeTemplateActionDefinition)
- decisionTreeTemplateActionDefinitionName (inherited from DecisionTreeTemplateActionDefinition), one or many
- decisionTreeTemplateActionDefinitionDescription (inherited from DecisionTreeTemplateActionDefinition), zero, one or many

3.8.3.7.2 Associations

This class has the following associations:

- A directed followedBy association with one instance of DecisionTreeTemplateFollowOnItem

3.8.3.7.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.8.3.8 DecisionTreeTemplateQuestionDefinition

DecisionTreeTemplateQuestionDefinition is a <<class>> that specifies a question that result in a YES or NO answer.

3.8.3.8.1 Attribute(s)

This class has the following attributes:

- decisionTreeTemplateQuestionDefinitionIdentifier
- decisionTreeTemplateQuestionDefinitionName, one or many
- decisionTreeTemplateQuestionDefinitionDescription, zero, one or many

3.8.3.8.2 Associations

This class has the following associations:

- A directed yesAnswer association with one instance of DecisionTreeTemplateFollowOnItem
- A directed noAnswer association with one instance of DecisionTreeTemplateFollowOnItem

3.8.3.8.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.8.3.8.4 Selects

This class is a member of the following <<select>> interfaces:

- DecisionTreeTemplateFollowOnItem
- DecisionTreeTemplateStartItem

3.8.3.9 DecisionTreeTemplateRevision

DecisionTreeTemplateRevision is a <<class>> representing an iteration applied to a DecisionTreeTemplate.

3.8.3.9.1 Attribute(s)

This class has the following attributes:

- decisionTreeTemplateRevisionIdentifier
- decisionTreeTemplateAnalysisDomain, zero, one or many
- decisionTreeTemplateDescription, zero, one or many
- decisionTreeTemplateRevisionRationale, zero, one or many
- decisionTreeTemplateRevisionDate, zero or one
- decisionTreeTemplateRevisionStatus, zero or one

3.8.3.9.2 Associations

This class has the following associations:

- An aggregate association with one or many instances of DecisionTreeTemplateActionDefinition
- An aggregate association with one or many instances of DecisionTreeTemplateQuestionDefinition
- A directed startItem association with one instance of DecisionTreeTemplateStartItem

3.8.3.9.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.8.3.10 DecisionTreeTemplateStartItem

DecisionTreeTemplateStartItem is a <<select>> interface that identifies items which can be selected as the starting point for the decision process.

3.8.3.10.1 Class members

This <<select>> interface includes the following class members:

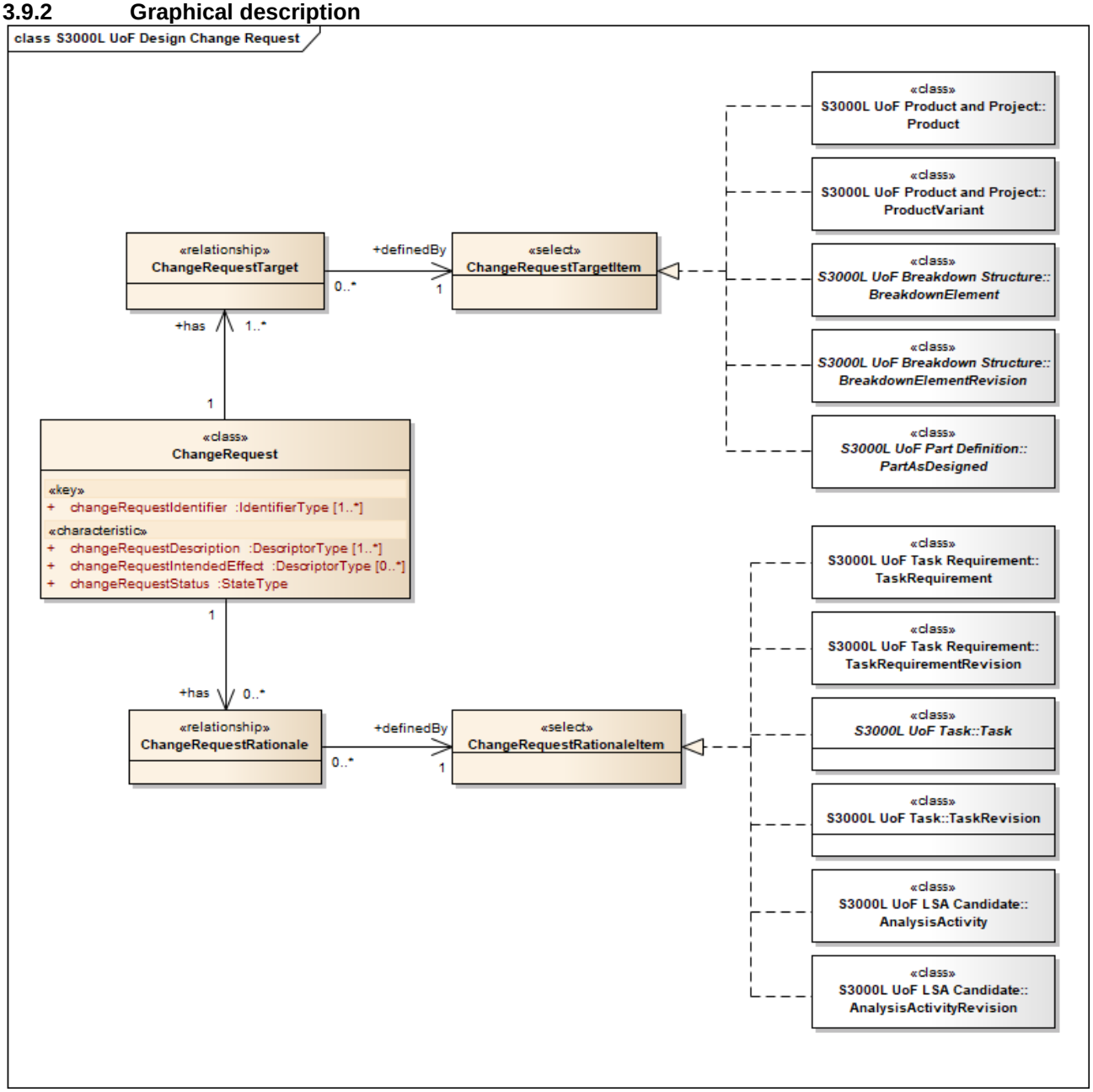
- DecisionTreeTemplateFurtherAnalysisActionDefinition
- DecisionTreeTemplateQuestionDefinition

3.9 S3000L UoF Design Change Request

3.9.1 Description

The Design Change Request UoF provides the capability to identify proposed changes against items including the rationale for this change.

3.9.2 Graphical description



ICN-B6865-S3000L0283-001-01

Fig 11 S3000L UoF Design Change Request

3.9.3 Class definition

3.9.3.1 ChangeRequest

`ChangeRequest` is a `<<class>>` that represents a formal proposal for a modification to a configuration item upon a given baseline.

Note

Typical configuration items are eg, `Product`, `PartAsDesigned`, and `BreakdownElement`.

3.9.3.1.1 Attribute(s)

This class has the following attributes:

- changeRequestIdentifier, one or many
- changeRequestDescription, one or many
- changeRequestIntendedEffect, zero, one or many
- changeRequestStatus

3.9.3.1.2 Associations

This class has the following associations:

- A directed has association with one or many instances of ChangeRequestTarget
- A directed has association with zero, one or many instances of ChangeRequestRationale

3.9.3.1.3 Implementations

This class implements the following <<extend>> interfaces:

- DigitalFileReferencingItem. Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.9.3.2 ChangeRequestRationale

ChangeRequestRationale is a <<relationship>> that associates a ChangeRequest with a ChangeRequestRationaleItem.

3.9.3.2.1 Associations

This class has the following associations:

- A directed definedby association with one instance of ChangeRequestRationaleItem

3.9.3.2.2 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.9.3.3 ChangeRequestRationaleItem

ChangeRequestRationaleItem is a <<select>> interface that identifies analysis items which can support a ChangeRequest.

3.9.3.3.1 Class members

This <<select>> interface includes the following class members:

- AnalysisActivity. Refer to Para 3.21
- AnalysisActivityRevision. Refer to Para 3.21
- Task. Refer to Para 3.36
- TaskRequirement. Refer to Para 3.37
- TaskRequirementRevision. Refer to Para 3.37

- TaskRevision. Refer to Para 3.36

3.9.3.4 ChangeRequestTarget

ChangeRequestTarget is a <<relationship>> that associates a ChangeRequest with a ChangeRequestTargetItem.

3.9.3.4.1 Associations

This class has the following associations:

- A directed definedBy association with one instance of ChangeRequestTargetItem

3.9.3.4.2 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.9.3.5 ChangeRequestTargetItem

ChangeRequestTargetItem is a <<select>> interface that identifies items which can be the subject for a ChangeRequest.

3.9.3.5.1 Class members

This <<select>> interface includes the following class members:

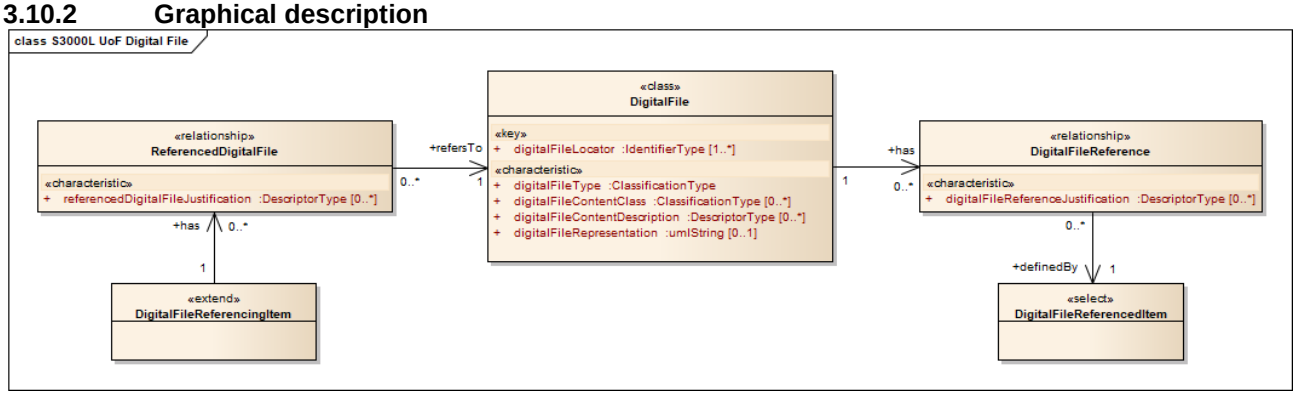
- BreakdownElement. Refer to Para 3.3
- BreakdownElementRevision. Refer to Para 3.3
- PartAsDesigned. Refer to Para 3.25
- Product. Refer to Para 3.27
- ProductVariant. Refer to Para 3.27

3.10 S3000L UoF Digital File

3.10.1 Description

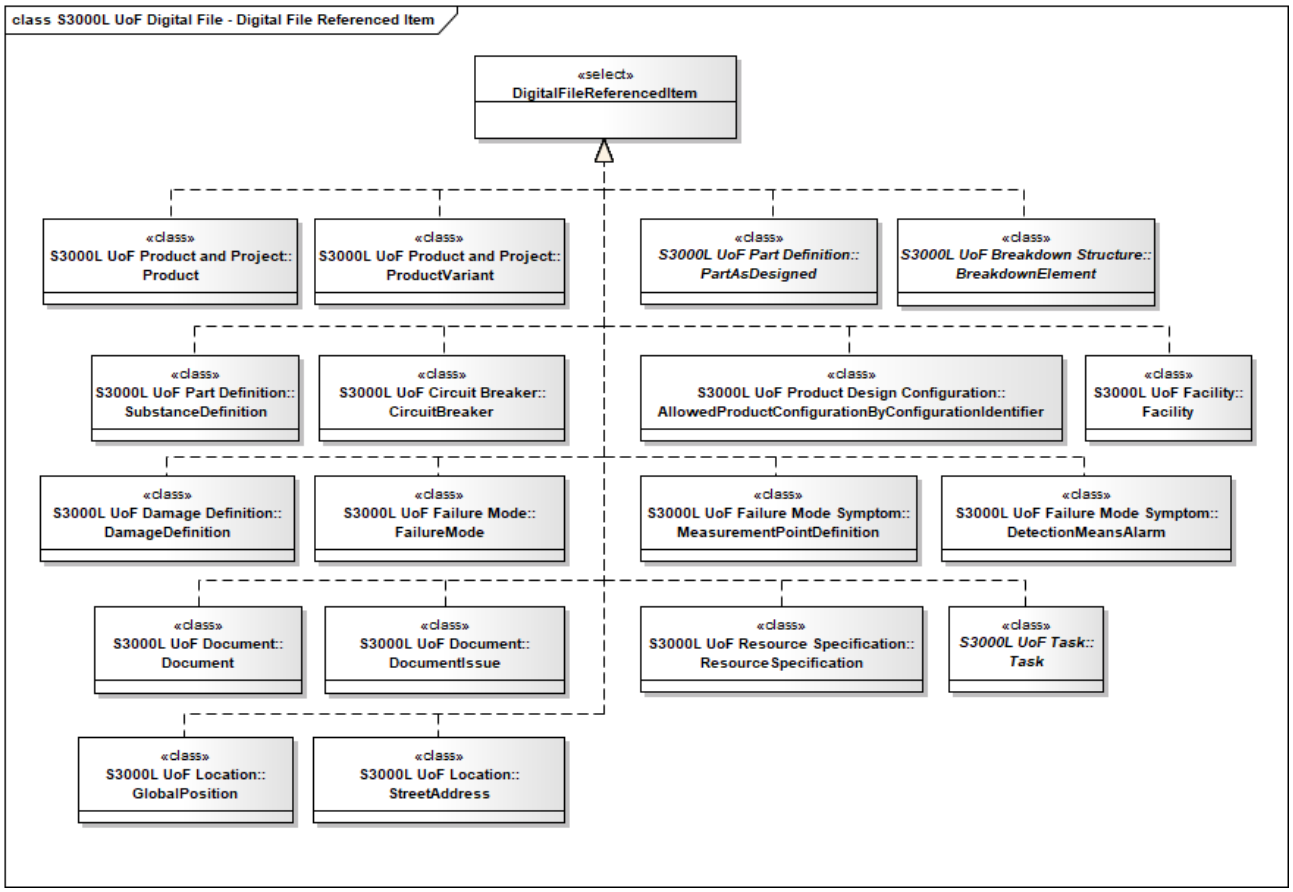
The Digital File UoF provides the capability to both reference a digital file from the exchanged data as well as to exchange the digital file itself.

3.10.2 Graphical description



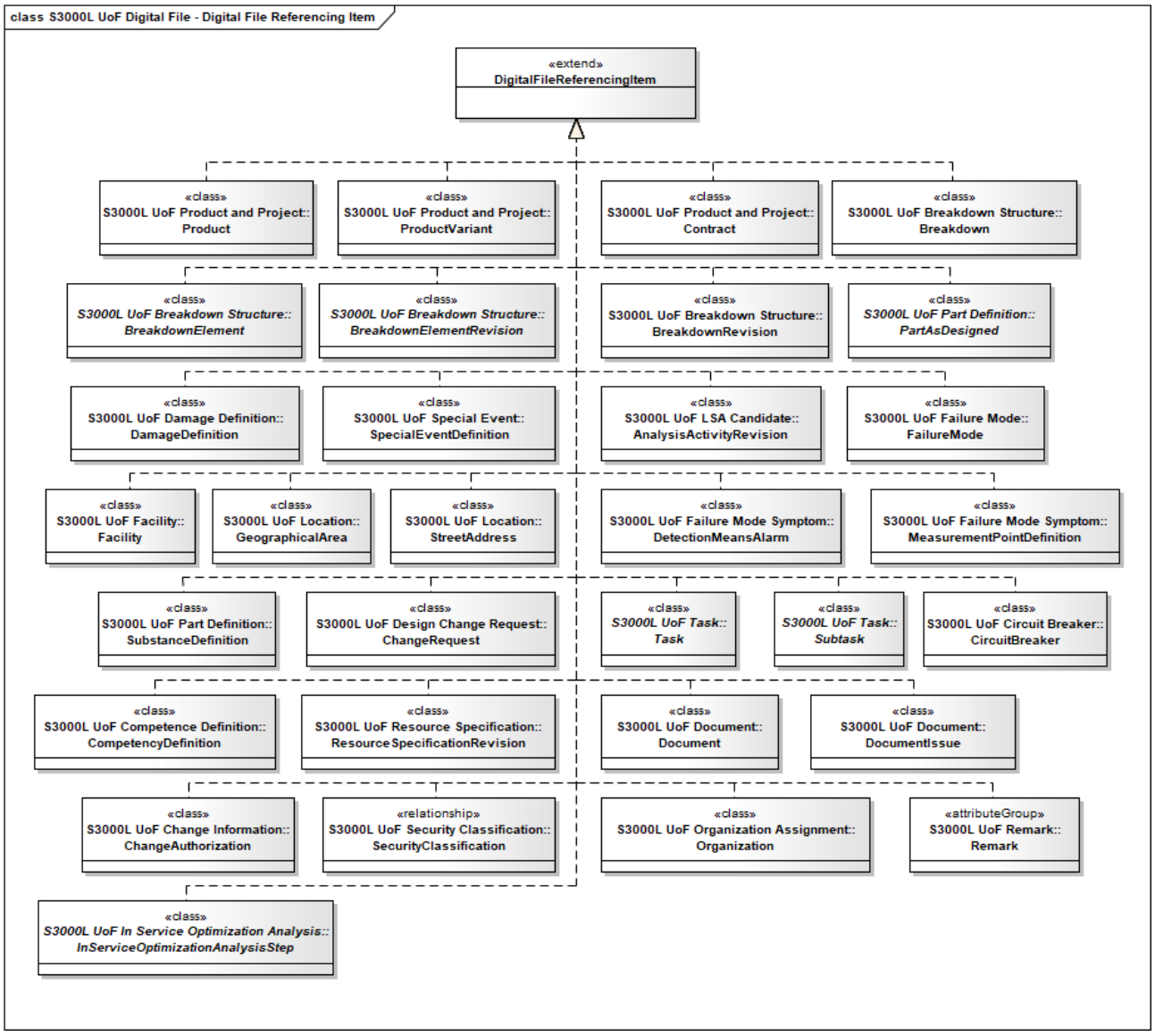
ICN-B6865-S3000L0292-001-01

Fig 12 S3000L UoF Digital File



ICN-B6865-S3000L0293-001-01

Fig 13 S3000L UoF Digital File - Digital File Referenced Item



ICN-B6865-S3000L0294-001-01

Fig 14 S3000L UoF Digital File - Digital File Referencing Item

3.10.3 Class definition

3.10.3.1 DigitalFile

`DigitalFile` is a `<<class>>` that provides the identification of data stored on an electronic device that can be interpreted by a computer.

3.10.3.1.1 Attribute(s)

This class has the following attributes:

- `digitalFileLocator`, one or many
- `digitalFileType`
- `digitalFileContentClass`, zero, one or many
- `digitalFileContentDescription`, zero, one or many
- `digitalFileRepresentation`, zero or one

3.10.3.1.2 Associations

This class has the following associations:

- A directed has association with zero, one or many instances of DigitalFileReference

3.10.3.1.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem. Refer to Para 3.33

3.10.3.2 DigitalFileReference

DigitalFileReference is a <<relationship>> that allows a DigitalFile to reference a DigitalFileReferencedItem.

3.10.3.2.1 Attribute(s)

This class has the following attributes:

- digitalFileReferenceJustification, zero, one or many

3.10.3.2.2 Associations

This class has the following associations:

- A directed definedBy association with one instance of DigitalFileReferencedItem

3.10.3.2.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.10.3.3 DigitalFileReferencedItem

DigitalFileReferencedItem is a <<select>> interface that identifies an item which in some way is associated with the content of the DigitalFile.

3.10.3.3.1 Class members

This <<select>> interface includes the following class members:

- AllowedProductConfigurationByConfigurationIdentifier. Refer to Para 3.28
- BreakdownElement. Refer to Para 3.3
- CircuitBreaker. Refer to Para 3.5
- DamageDefinition. Refer to Para 3.7
- DetectionMeansAlarm. Refer to Para 3.16
- Document. Refer to Para 3.11
- DocumentIssue. Refer to Para 3.11
- Facility. Refer to Para 3.13

- FailureMode. Refer to Para 3.14
- GlobalPosition. Refer to Para 3.19
- MeasurementPointDefinition. Refer to Para 3.16
- PartAsDesigned. Refer to Para 3.25
- Product. Refer to Para 3.27
- ProductVariant. Refer to Para 3.27
- ResourceSpecification. Refer to Para 3.32
- StreetAddress. Refer to Para 3.19
- SubstanceDefinition. Refer to Para 3.25
- Task. Refer to Para 3.36

3.10.3.4 DigitalFileReferencingItem

DigitalFileReferencingItem is an <<extend>> interface that provides its associated data model to those classes that implement it.

3.10.3.4.1 Class members

Classes that implement the DigitalFileReferencingItem <<extend>> are:

- AnalysisActivityRevision. Refer to Para 3.21
- Breakdown. Refer to Para 3.3
- BreakdownElement. Refer to Para 3.3
- BreakdownElementRevision. Refer to Para 3.3
- BreakdownRevision. Refer to Para 3.3
- ChangeAuthorization. Refer to Para 3.4
- ChangeRequest. Refer to Para 3.9
- CircuitBreaker. Refer to Para 3.5
- CompetencyDefinition. Refer to Para 3.6
- Contract. Refer to Para 3.27
- DamageDefinition. Refer to Para 3.7
- DetectionMeansAlarm. Refer to Para 3.16
- Document. Refer to Para 3.11
- DocumentIssue. Refer to Para 3.11
- Facility. Refer to Para 3.13
- FailureMode. Refer to Para 3.14
- GeographicalArea. Refer to Para 3.19
- InServiceOptimizationAnalysisStep. Refer to Para 3.18
- MeasurementPointDefinition. Refer to Para 3.16
- Organization. Refer to Para 3.24
- PartAsDesigned. Refer to Para 3.25
- Product. Refer to Para 3.27
- ProductVariant. Refer to Para 3.27
- Remark. Refer to Para 3.31
- ResourceSpecificationRevision. Refer to Para 3.32
- SecurityClassification. Refer to Para 3.33
- SpecialEventDefinition. Refer to Para 3.35
- StreetAddress. Refer to Para 3.19
- SubstanceDefinition. Refer to Para 3.25
- Subtask. Refer to Para 3.36
- Task. Refer to Para 3.36

3.10.3.4.2 Associations

The DigitalFileReferencingItem <<extend>> interface has the following associations:

- A directed has association with zero, one or many instances of ReferencedDigitalFile

3.10.3.5 ReferencedDigitalFile

ReferencedDigitalFile is a <<relationship> that allows an item to refer to a DigitalFile.

3.10.3.5.1 Attribute(s)

This class has the following attributes:

- referencedDigitalFileJustification, zero, one or many

3.10.3.5.2 Associations

This class has the following associations:

- A directed refersTo association with one instance of DigitalFile

3.10.3.5.3 Implementations

This class implements the following <<extend>> interfaces:

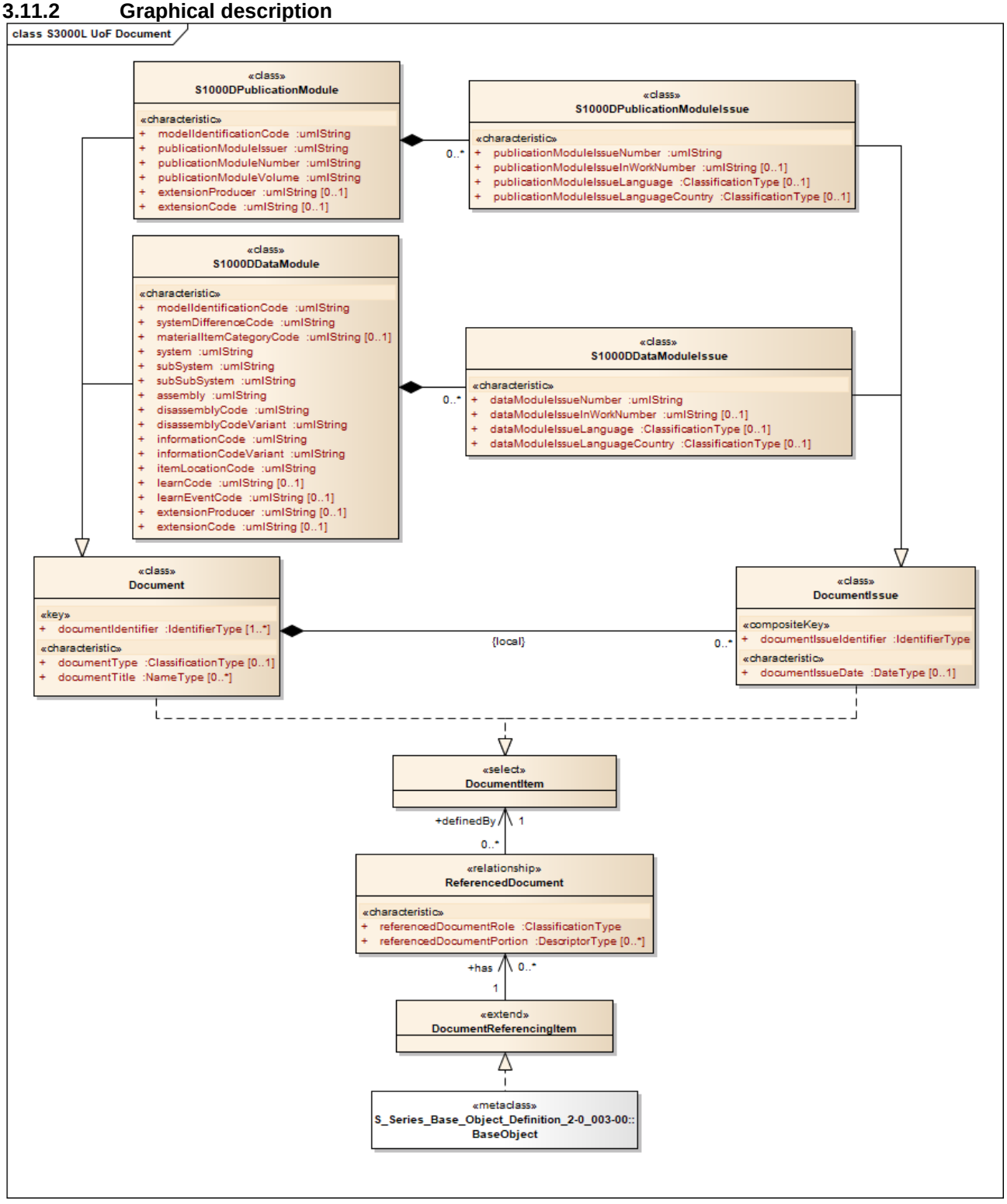
- ApplicabilityStatementItem. Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.11 S3000L UoF Document

3.11.1 Description

The Document UoF provides the capability to identify a physical document or a digital file and their associated metadata.

3.11.2 Graphical description



ICN-B6865-S3000L0233-004-01

Fig 15 S3000L UoF Document

3.11.3 Class definition

3.11.3.1 Document

Document is a <<class>> that represents a compiled set of information that serves a purpose.

Example(s)

- Drawing
- Manual
- Report

3.11.3.1.1 Attribute(s)

This class has the following attributes:

- documentIdentifier, one or many
- documentType, zero or one
- documentTitle, zero, one or many

3.11.3.1.2 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of DocumentIssue

3.11.3.1.3 Implementations

This class implements the following <<extend>> interfaces:

- DigitalFileReferencingItem. Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject)
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem. Refer to Para 3.33

3.11.3.2 DocumentIssue

DocumentIssue is a <<class>> that represents a specific release of a Document

3.11.3.2.1 Attribute(s)

This class has the following attributes:

- documentIssueIdentifier
- documentIssueDate, zero or one

3.11.3.2.2 Implementations

This class implements the following <<extend>> interfaces:

- ChangeControlledItem. Refer to Para 3.4
- DigitalFileReferencingItem. Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject)
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.11.3.3 DocumentItem

DocumentItem is a <<select>> interface that identifies items which can be selected as Document.

3.11.3.3.1 Class members

This <<select>> interface includes the following class members:

- Document
- DocumentIssue

3.11.3.4 DocumentReferencingItem

DocumentReferencingItem is an <<extend>> interface that provides its associated data model to those classes that implement it.

3.11.3.4.1 Class members

Classes that implement the DocumentReferencingItem <<extend>> interface are:

- AllocatedTaskLocation. Refer to Para 3.39
- AllowedProductConfigurationByConfigurationIdentifier. Refer to Para 3.28
- AnalysisActivity. Refer to Para 3.21
- AnalysisActivityRevision. Refer to Para 3.21
- ApplicabilityStatement. Refer to Para 3.2
- AuthorityToOperate. Refer to Para 3.28
- Breakdown. Refer to Para 3.3
- BreakdownElement. Refer to Para 3.3
- BreakdownElementRevision. Refer to Para 3.3
- BreakdownElementUsageInBreakdown. Refer to Para 3.3
- BreakdownRevision. Refer to Para 3.3
- ChangeAuthorization. Refer to Para 3.4
- ChangeRequest. Refer to Para 3.9
- CircuitBreaker. Refer to Para 3.5
- CircuitBreakerSetting. Refer to Para 3.36
- CircuitBreakerSettings. Refer to Para 3.36
- CompetencyDefinition. Refer to Para 3.6
- ConditionInstance. Refer to Para 3.2
- ConditionType. Refer to Para 3.2
- ConditionTypeAssertMember. Refer to Para 3.2
- Contract. Refer to Para 3.27
- Country. Refer to Para 3.19
- DamageDefinition. Refer to Para 3.7
- DecisionTreeTemplate. Refer to Para 3.8
- DecisionTreeTemplateActionDefinition. Refer to Para 3.8
- DecisionTreeTemplateQuestionDefinition. Refer to Para 3.8
- DecisionTreeTemplateRevision. Refer to Para 3.8
- DetectionMeansAlarm. Refer to Para 3.16
- DigitalFile. Refer to Para 3.10
- Document. Refer to Para 3.11
- DocumentIssue. Refer to Para 3.11
- EnvironmentDefinition. Refer to Para 3.12
- EnvironmentDefinitionCharacteristic. Refer to Para 3.12

- EnvironmentDefinitionRevision. Refer to Para 3.12
- EvaluationCriteria. Refer to Para 3.2
- Facility. Refer to Para 3.13
- FailureMode. Refer to Para 3.14
- FailureModeAnalysis. Refer to Para 3.14
- FailureModeAnalysisRevision. Refer to Para 3.14
- FailureModeCause. Refer to Para 3.14
- FailureModeEffect. Refer to Para 3.14
- FailureModeSymptomsSignature. Refer to Para 3.16
- GeographicalArea. Refer to Para 3.19
- GlobalPosition. Refer to Para 3.19
- InServiceOptimizationAnalysis. Refer to Para 3.18
- InServiceOptimizationAnalysisRevision. Refer to Para 3.18
- InServiceOptimizationAnalysisStep. Refer to Para 3.18
- LSAFailureModeGroup. Refer to Para 3.22
- MaintenanceLevel. Refer to Para 3.29
- MeasurementPointDefinition. Refer to Para 3.16
- Message. Refer to Para 3.23
- OperatingLocationType. Refer to Para 3.29
- Organization. Refer to Para 3.24
- PartAsDesigned. Refer to Para 3.25
- PartAsDesignedPartsList. Refer to Para 3.25
- PartAsDesignedPartsListEntry. Refer to Para 3.25
- PartAsDesignedPartsListRevision. Refer to Para 3.25
- PerformanceParameter. Refer to Para 3.26
- PerformanceParameterRevision. Refer to Para 3.26
- Product. Refer to Para 3.27
- ProductUsagePhase. Refer to Para 3.30
- ProductVariant. Refer to Para 3.27
- Project. Refer to Para 3.27
- RepeatDefinition. Refer to Para 3.40
- ResourceSpecification. Refer to Para 3.32
- ResourceSpecificationRevision. Refer to Para 3.32
- SecurityClass. Refer to Para 3.33
- Skill. Refer to Para 3.6
- SkillLevel. Refer to Para 3.6
- SpecialEventDefinition. Refer to Para 3.35
- StreetAddress. Refer to Para 3.19
- SubstanceDefinition. Refer to Para 3.25
- Subtask. Refer to Para 3.36
- Task. Refer to Para 3.36
- TaskRequirement. Refer to Para 3.37
- TaskRequirementRevision. Refer to Para 3.37
- TaskResource. Refer to Para 3.38
- TaskRevision. Refer to Para 3.36
- ThresholdDefinition. Refer to Para 3.40
- TimeLimit. Refer to Para 3.40
- Trade. Refer to Para 3.6

- WarningCautionNote. Refer to Para 3.36

3.11.3.4.2 Associations

The DocumentReferencingItem <<extend>> interface has the following associations:

- A directed has association with zero, one or many instances of ReferencedDocument

3.11.3.5 ReferencedDocument

ReferencedDocument is a <<relationship>> where one DocumentReferencingItem relates to a DocumentItem.

3.11.3.5.1 Attribute(s)

This class has the following attributes:

- referencedDocumentRole
- referencedDocumentPortion, zero, one or many

3.11.3.5.2 Associations

This class has the following associations:

- A directed definedBy association with one instance of DocumentItem

3.11.3.5.3 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject)
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.11.3.6 S1000DDataModule

S1000DDataModule is a Document that is written in accordance with an S1000D schema.

3.11.3.6.1 Attribute(s)

This class has the following attributes:

- documentIdentifier (inherited from Document), one or many
- documentType (inherited from Document), zero or one
- documentTitle (inherited from Document), zero, one or many
- modelIdentificationCode
- systemDifferenceCode
- materialItemCategoryCode, zero or one
- system
- subSystem
- subSubSystem
- assembly
- disassemblyCode
- disassemblyCodeVariant
- informationCode

- informationCodeVariant
- itemLocationCode
- learnCode, zero or one
- learnEventCode, zero or one
- extensionProducer, zero or one
- extensionCode, zero or one

3.11.3.6.2 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of S1000DDataModuleIssue

3.11.3.6.3 Implementations

This class implements the following <<extend>> interfaces:

- DigitalFileReferencingItem (inherited from Document). Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem (inherited from Document). Refer to Para 3.33

3.11.3.7 S1000DDataModuleIssue

S1000DDataModuleIssue is a DocumentIssue that identifies a specific issue of a data module produced in accordance with S1000D.

3.11.3.7.1 Attribute(s)

This class has the following attributes:

- documentIssueIdentifier (inherited from DocumentIssue)
- documentIssueDate (inherited from DocumentIssue), zero or one
- dataModuleIssueNumber
- dataModuleIssueInWorkNumber, zero or one
- dataModuleIssueLanguage, zero or one
- dataModuleIssueLanguageCountry, zero or one

3.11.3.7.2 Implementations

This class implements the following <<extend>> interfaces:

- ChangeControlledItem (inherited from DocumentIssue). Refer to Para 3.4
- DigitalFileReferencingItem (inherited from DocumentIssue). Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject)
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.11.3.8 S1000DPublicationModule

S1000DPublicationModule is a Document that identifies a publication published in accordance with S1000D.

3.11.3.8.1 Attribute(s)

This class has the following attributes:

- documentIdentifier (inherited from Document), one or many
- documentType (inherited from Document), zero or one
- documentTitle (inherited from Document), zero, one or many
- modelIdentificationCode
- publicationModuleIssuer
- publicationModuleNumber
- publicationModuleVolume
- extensionProducer, zero or one
- extensionCode, zero or one

3.11.3.8.2 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of S1000DPublicationModuleIssue

3.11.3.8.3 Implementations

This class implements the following <<extend>> interfaces:

- DigitalFileReferencingItem (inherited from Document). Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject)
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem (inherited from Document). Refer to Para 3.33

3.11.3.9 S1000DPublicationModuleIssue

S1000DPublicationModuleIssue is a DocumentIssue that identifies a specific issue of a publication module published in accordance with S1000D.

3.11.3.9.1 Attribute(s)

This class has the following attributes:

- documentIssueIdentifier (inherited from DocumentIssue)
- documentIssueDate (inherited from DocumentIssue), zero or one
- publicationModuleIssueNumber
- publicationModuleIssueInWorkNumber, zero or one
- publicationModuleIssueLanguage, zero or one
- publicationModuleIssueLanguageCountry, zero or one

3.11.3.9.2 Implementations

This class implements the following <<extend>> interfaces:

- ChangeControlledItem (inherited from DocumentIssue). Refer to Para 3.4
- DigitalFileReferencingItem (inherited from DocumentIssue). Refer to Para 3.10

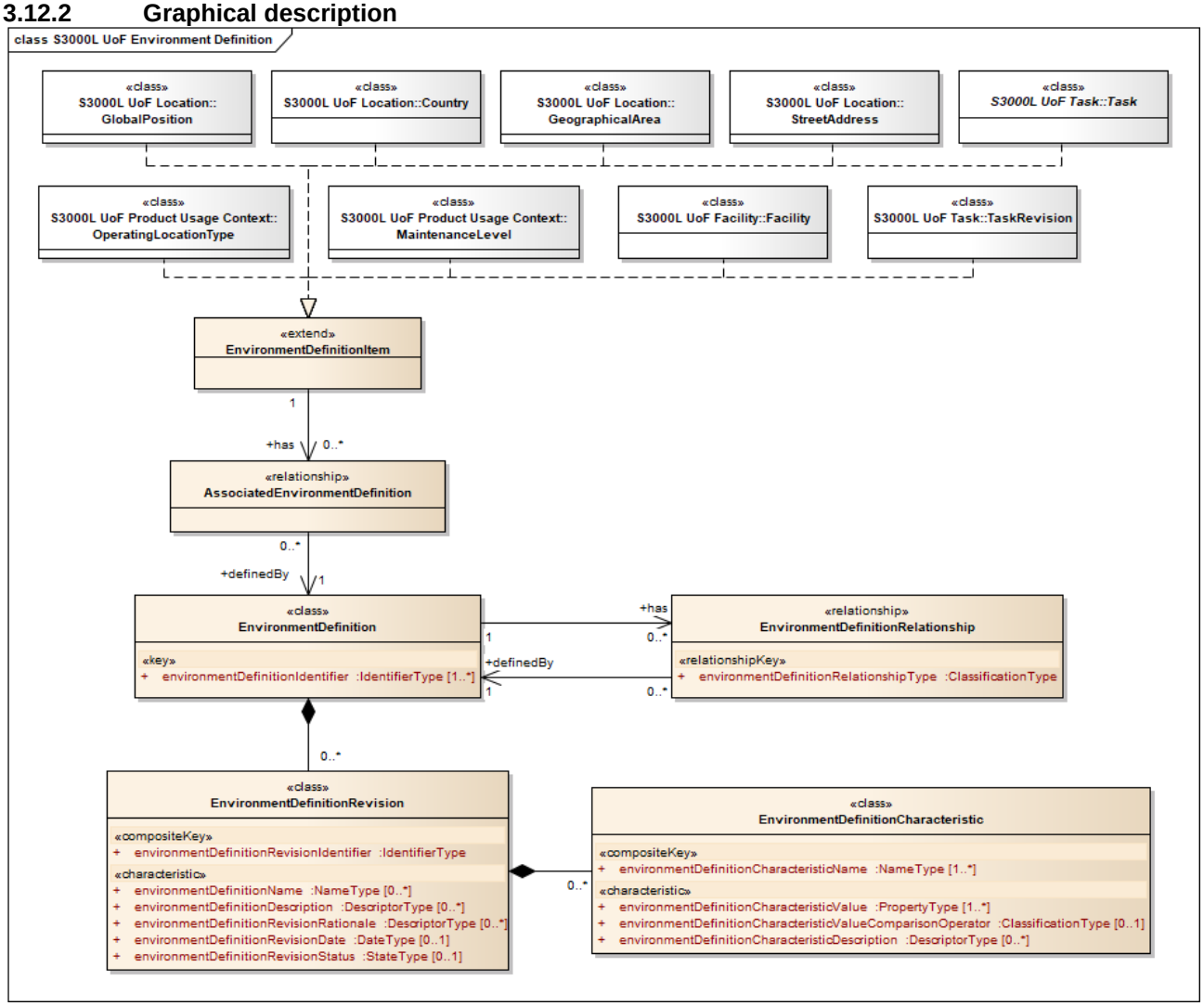
- DocumentReferencingItem (inherited from BaseObject)
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.12 S3000L UoF Environment Definition

3.12.1 Description

The Environment Definition UoF provides the capability to define circumstances, objects, events and/or conditions by which something can be surrounded and that influence the performance of an associated item.

3.12.2 Graphical description



ICN-B6865-S3000L0272-001-01

Fig 16 S3000L UoF Environment Definition

3.12.3 Class definition

3.12.3.1 AssociatedEnvironmentDefinition

AssociatedEnvironmentDefinition is a <<relationship>> that associates an EnvironmentDefinitionItem with an EnvironmentDefinition relevant to its existence, operation and/or support.

3.12.3.1.1 Associations

This class has the following associations:

- A directed definedBy association with one instance of EnvironmentDefinition

3.12.3.2 EnvironmentDefinition

EnvironmentDefinition is a <<class>> that specifies the circumstances, objects, events and/or conditions by which something can be surrounded and that influence the performance of an associated item.

3.12.3.2.1 Attribute(s)

This class has the following attributes:

- environmentDefinitionIdentifier, one or many

3.12.3.2.2 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of EnvironmentDefinitionRevision
- A directed has association with zero, one or many instances of EnvironmentDefinitionRelationship

3.12.3.2.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.12.3.3 EnvironmentDefinitionCharacteristic

EnvironmentDefinitionCharacteristic is a <<class>> that represents a measurable or observable characteristic for a circumstance, object, event or condition that is significant to the EnvironmentDefinition.

3.12.3.3.1 Attribute(s)

This class has the following attributes:

- environmentDefinitionCharacteristicName, one or many
- environmentDefinitionCharacteristicValue, one or many
- environmentDefinitionCharacteristicValueComparisonOperator, zero or one
- environmentDefinitionCharacteristicDescription, zero, one or many

3.12.3.3.2 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.12.3.4 EnvironmentDefinitionItem

EnvironmentDefinitionItem is an <<extend>> interface that provides its associated data model to those classes that implement it.

3.12.3.4.1 Class members

Classes that implement the EnvironmentDefinitionItem <<extend>> interface are:

- Country. Refer to Para 3.19
- Facility. Refer to Para 3.13
- GeographicalArea. Refer to Para 3.19
- GlobalPosition. Refer to Para 3.19
- MaintenanceLevel. Refer to Para 3.29
- OperatingLocationType. Refer to Para 3.29
- StreetAddress. Refer to Para 3.19
- Task. Refer to Para 3.36
- TaskRevision. Refer to Para 3.36

3.12.3.4.2 Associations

The EnvironmentDefinitionItem <<extend>> interface has the following associations:

- A directed has association with zero, one or many instances of AssociatedEnvironmentDefinition

3.12.3.5 EnvironmentDefinitionRelationship

EnvironmentDefinitionRelationship is a <<relationship>> where one EnvironmentDefinition relates to another EnvironmentDefinition.

3.12.3.5.1 Attribute(s)

This class has the following attributes:

- environmentDefinitionRelationshipType

3.12.3.5.2 Associations

This class has the following associations:

- A directed definedBy association with one instance of EnvironmentDefinition

3.12.3.6 EnvironmentDefinitionRevision

EnvironmentDefinitionRevision is a <<class>> representing an iteration applied to an EnvironmentDefinition.

3.12.3.6.1 Attribute(s)

This class has the following attributes:

- environmentDefinitionRevisionIdentifier
- environmentDefinitionName, zero, one or many
- environmentDefinitionDescription, zero, one or many
- environmentDefinitionRevisionRationale, zero, one or many
- environmentDefinitionRevisionDate, zero or one
- environmentDefinitionRevisionStatus, zero or one

3.12.3.6.2 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of EnvironmentDefinitionCharacteristic

3.12.3.6.3 Implementations

This class implements the following <<extend>> interfaces:

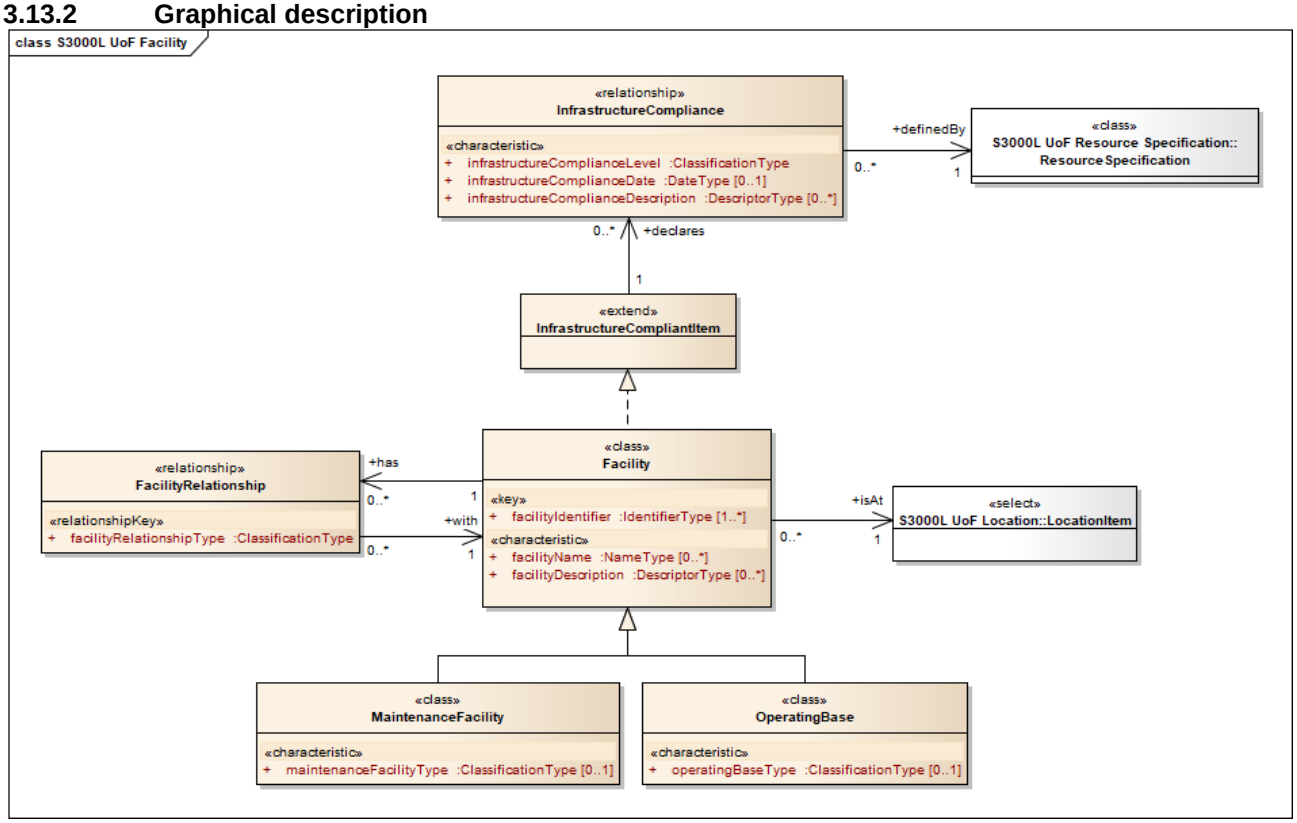
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.13 S3000L UoF Facility

3.13.1 Description

The Facility UoF provides the capability to identify a facility and to declare to what extent it fulfills a set of identified infrastructure requirements.

3.13.2 Graphical description



ICN-B6865-S3000L0270-001-01

Fig 17 S3000L UoF Facility

3.13.3 Class definition

3.13.3.1 Facility

`Facility` is a <<class> that represents a physically limited infrastructure which exists, or is intended to be built or installed, and is established to serve a particular purpose.

3.13.3.1.1 Attribute(s)

This class has the following attributes:

- `facilityIdentifier`, one or many
- `facilityName`, zero, one or many
- `facilityDescription`, zero, one or many

3.13.3.1.2 Associations

This class has the following associations:

- A directed `has` association with zero, one or many instances of `FacilityRelationship`
- A directed `isAt` association with one instance of `LocationItem`. Refer to Para 3.19

3.13.3.1.3 Implementations

This class implements the following `<<extend>>` interfaces:

- `DigitalFileReferencingItem`. Refer to Para 3.10
- `DocumentReferencingItem` (inherited from `BaseObject`). Refer to Para 3.11
- `EnvironmentDefinitionItem`. Refer to Para 3.12
- `InfrastructureCompliantItem`. Refer to Para 3.13

- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem. Refer to Para 3.33

3.13.3.2 FacilityRelationship

FacilityRelationship is a <<relationship>> where one Facility relates to another Facility.

3.13.3.2.1 Attribute(s)

This class has the following attributes:

- facilityRelationshipType

3.13.3.2.2 Associations

This class has the following associations:

- A directed with association with one instance of Facility

3.13.3.2.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.13.3.3 InfrastructureCompliance

InfrastructureCompliance is a <<relationship>> that documents how the InfrastructureCompliantItem fulfills requirements stated in the associated ResourceSpecification.

3.13.3.3.1 Attribute(s)

This class has the following attributes:

- infrastructureComplianceLevel
- infrastructureComplianceDate, zero or one
- infrastructureComplianceDescription, zero, one or many

3.13.3.3.2 Associations

This class has the following associations:

- A directed definedBy association with one instance of ResourceSpecification. Refer to Para 3.32

3.13.3.3.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24

- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.13.3.4 InfrastructureCompliantItem

InfrastructureCompliantItem is an <<extend>> interface that provides its associated data model to those classes that implement it.

3.13.3.4.1 Class members

Classes that implement the InfrastructureCompliantItem <<extend>> interface are:

- Facility

3.13.3.4.2 Associations

The InfrastructureCompliantItem <<extend>> interface has the following associations:

- A directed association with zero, one or many instances of InfrastructureCompliance

3.13.3.5 MaintenanceFacility

MaintenanceFacility is a Facility that is mainly established for providing product support.

3.13.3.5.1 Attribute(s)

This class has the following attributes:

- facilityIdentifier (inherited from Facility), one or many
- facilityName (inherited from Facility), zero, one or many
- facilityDescription (inherited from Facility), zero, one or many
- maintenanceFacilityType, zero or one

3.13.3.5.2 Associations

This class has the following associations:

- A directed has association with zero, one or many instances of FacilityRelationship (inherited from Facility)
- A directed isAt association with one instance of LocationItem (inherited from Facility)
- A directed maintenanceCapability association with zero or one instances of MaintenanceLevel. Refer to Para 3.29

3.13.3.5.3 Implementations

This class implements the following <<extend>> interfaces:

- DigitalFileReferencingItem (inherited from Facility). Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- EnvironmentDefinitionItem (inherited from Facility). Refer to Para 3.12
- InfrastructureCompliantItem (inherited from Facility). Refer to Para 3.13
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

- SecurityClassificationItem (inherited from Facility). Refer to Para 3.33

3.13.3.6 OperatingBase

OperatingBase is a Facility that is mainly established for providing support for operations.

3.13.3.6.1 Attribute(s)

This class has the following attributes:

- facilityIdentifier (inherited from Facility), one or many
- facilityName (inherited from Facility), zero, one or many
- facilityDescription (inherited from Facility), zero, one or many
- operatingBaseType, zero or one

3.13.3.6.2 Associations

This class has the following associations:

- A directed has association with zero, one or many instances of FacilityRelationship (inherited from Facility)
- A directed isAt association with one instance of LocationItem (inherited from Facility)
- A directed operatingCapability association with zero or one instances of OperatingLocationType. Refer to Para 3.29

3.13.3.6.3 Implementations

This class implements the following <<extend>> interfaces:

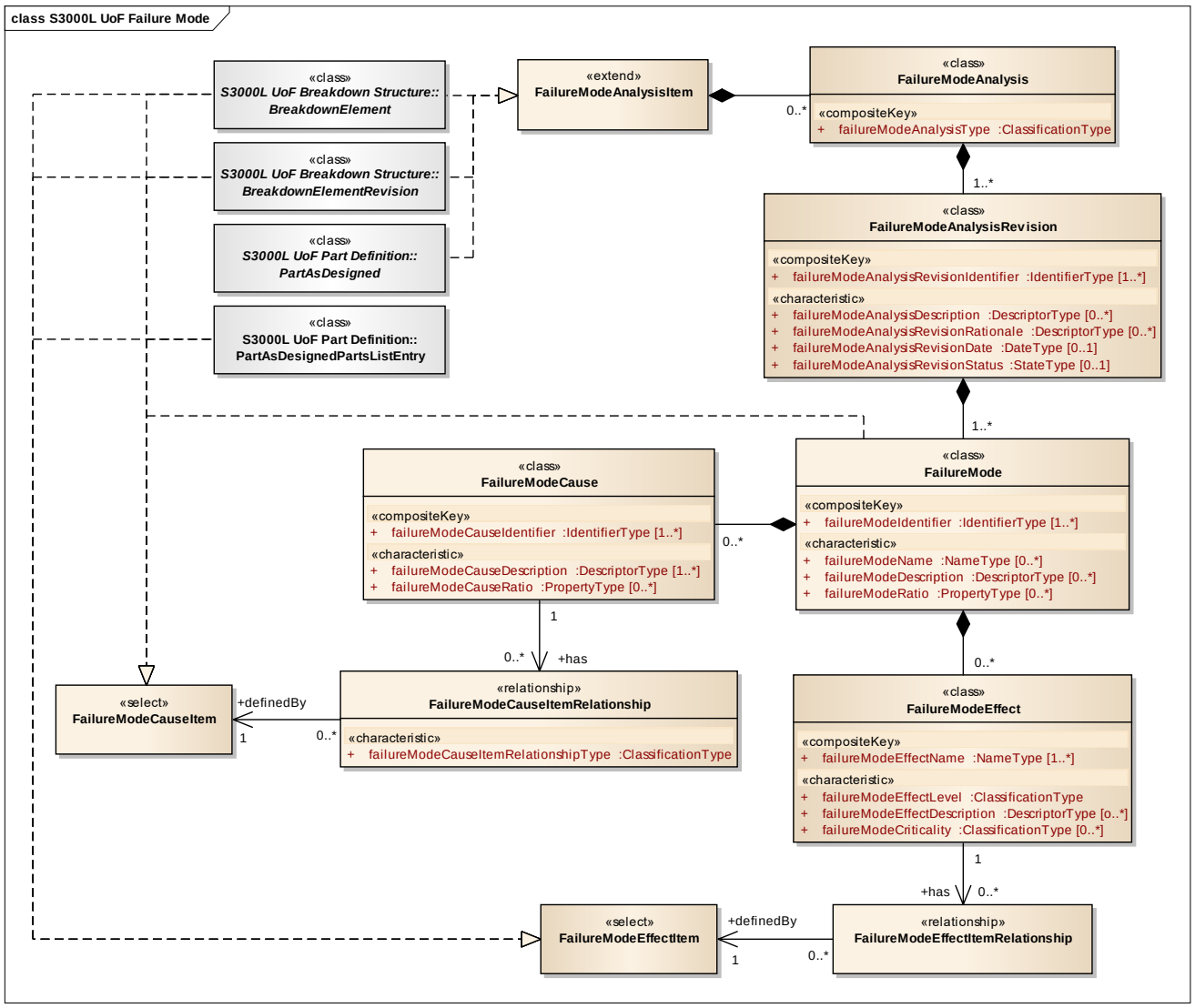
- DigitalFileReferencingItem (inherited from Facility). Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- EnvironmentDefinitionItem (inherited from Facility). Refer to Para 3.12
- InfrastructureCompliantItem (inherited from Facility). Refer to Para 3.13
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem (inherited from Facility). Refer to Para 3.33

3.14 S3000L UoF Failure Mode

3.14.1 Description

The Failure Mode UoF provides the capability to define the failure modes, their causes and effects.

3.14.2 Graphical description



ICN-B6865-S3000L0276-001-01

Fig 18 S3000L UoF Failure Mode

3.14.3 Class definition

3.14.3.1 FailureMode

`FailureMode` is a `<<class>>` that defines a functional consequence of an unacceptable `state` of the `FailureModeAnalysisItem`.

Example(s)

– No output from electrical circuit.

3.14.3.1.1 Attribute(s)

This class has the following attributes:

– `failureModeIdentifier`, one or many
– `failureModeName`, zero, one or many
– `failureModeDescription`, zero, one or many
– `failureModeRatio`, zero, one or many

3.14.3.1.2 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of FailureModeCause
- An aggregate association with zero, one or many instances of FailureModeEffect

3.14.3.1.3 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- DigitalFileReferencingItem. Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- FailureModeIsolationItem. Refer to Para 3.15
- FailureModeSymptomsSignatureItem. Refer to Para 3.16
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.14.3.1.4 Selects

This class is a member of the following <<select>> interfaces:

- DigitalFileReferencedItem. Refer to Para 3.10
- FailureModeCauseItem
- LSAFailureModeGroupItem. Refer to Para 3.22

3.14.3.2 FailureModeAnalysis

FailureModeAnalysis is a <<class>> that represents failure modes, effects and criticality identified for the associated FailureModeAnalysisItem.

3.14.3.2.1 Attribute(s)

This class has the following attributes:

- failureModeAnalysisType

3.14.3.2.2 Associations

This class has the following associations:

- An aggregate association with one or many instances of FailureModeAnalysisRevision

3.14.3.2.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.14.3.3 FailureModeAnalysisItem

FailureModeAnalysisItem is an <<extend>> interface that provides its associated data model to those classes that can have an associated FailureModeAnalysis.

3.14.3.3.1 Class members

Classes that implement the FailureModeAnalysisItem <<extend>> are:

- BreakdownElement. Refer to Para 3.3
- BreakdownElementRevision. Refer to Para 3.3
- PartAsDesigned. Refer to Para 3.25

3.14.3.3.2 Associations

The FailureModeAnalysisItem <<extend>> interface has the following associations:

- An aggregate association with zero, one or many. instances of FailureModeAnalysis

3.14.3.4 FailureModeAnalysisRevision

FailureModeAnalysisRevision is a <<class>> representing an iteration applied to a FailureModeAnalysis.

3.14.3.4.1 Attribute(s)

This class has the following attributes:

- failureModeAnalysisRevisionIdentifier, one or many
- failureModeAnalysisDescription, zero, one or many
- failureModeAnalysisRevisionRationale, zero, one or many
- failureModeAnalysisRevisionDate, zero or one
- failureModeAnalysisRevisionStatus, zero or one

3.14.3.4.2 Associations

This class has the following associations:

- An aggregate association with one or many instances of FailureMode

3.14.3.4.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.14.3.5 FailureModeCause

FailureModeCause is a <<class>> that specifies the physical or chemical process(es) that is the reason for the FailureMode.

3.14.3.5.1 Attribute(s)

This class has the following attributes:

- failureModeCauseIdentifier, one or many
- failureModeCauseDescription, one or many
- failureModeCauseRatio, zero, one or many

3.14.3.5.2 Associations

This class has the following associations:

- A directed has association with zero, one or many instances of FailureModeCauseItemRelationship

3.14.3.5.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.14.3.6 FailureModeCauseItem

FailureModeCauseItem is a <<select>> interface that identifies items which can be selected as being associated with a FailureModeCause.

3.14.3.6.1 Class members

This <<select>> interface includes the following class members:

- BreakdownElement. Refer to Para 3.3
- BreakdownElementRevision. Refer to Para 3.3
- FailureMode
- PartAsDesignedPartsListEntry. Refer to Para 3.25

3.14.3.7 FailureModeCauseItemRelationship

FailureModeCauseItemRelationship is a <<relationship>> where a FailureModeCause relates to the FailureModeCauseItem that in some way is associated with the FailureModeCause.

3.14.3.7.1 Attribute(s)

This class has the following attributes:

- failureModeCauseItemRelationshipType

3.14.3.7.2 Associations

This class has the following associations:

- A directed definedBy association with one instance of FailureModeCauseItem

3.14.3.7.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.14.3.8 FailureModeEffect

FailureModeEffect is a <<class>> that defines the consequences of an identified FailureMode on the operation, function, or status for the referred item.

3.14.3.8.1 Attribute(s)

This class has the following attributes:

- failureModeEffectName, one or many
- failureModeEffectLevel
- failureModeEffectDescription
- failureModeCriticality, zero, one or many

3.14.3.8.2 Associations

This class has the following associations:

- A directed has association with zero, one or many instances of FailureModeEffectItemRelationship

3.14.3.8.3 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.14.3.9 FailureModeEffectItem

FailureModeEffectItem is a <<select>> interface that identifies items which can be selected as being affected by a FailureMode.

3.14.3.9.1 Class members

This <<select>> interface includes the following class members:

- BreakdownElement. Refer to Para 3.3
- BreakdownElementRevision. Refer to Para 3.3
- PartAsDesignedPartsListEntry. Refer to Para 3.25

3.14.3.10 FailureModeEffectItemRelationship

FailureModeEffectItemRelationship is a <<relationship>> where a FailureModeEffect relates to the FailureModeEffectItem that is affected by the FailureMode.

3.14.3.10.1 Associations

This class has the following associations:

- A directed definedBy association with one instance of FailureModeEffectItem

3.14.3.10.2 Implementations

This class implements the following <<extend>> interfaces:

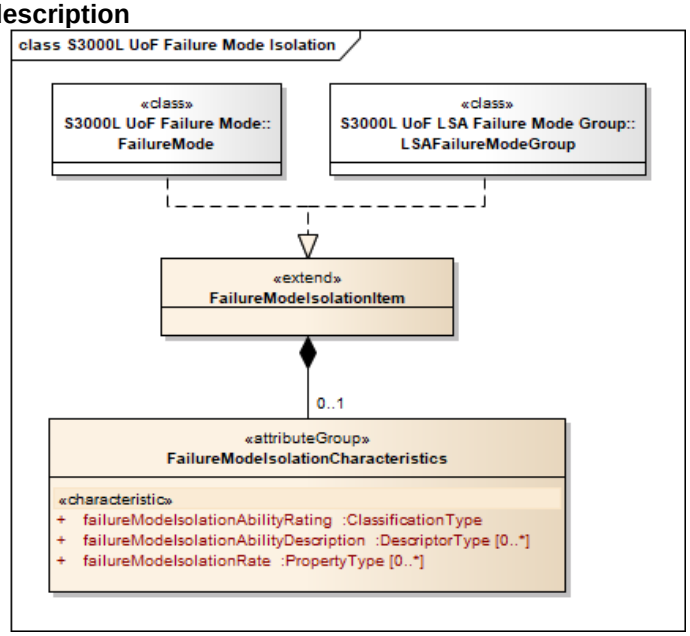
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.15 S3000L UoF Failure Mode Isolation

3.15.1 Description

The Failure Mode Isolation UoF provides the capability to collect data which describes the ability to determine that it the associated failure mode that have occurred.

3.15.2 Graphical description



ICN-B6865-S3000L0279-001-01

Fig 19 S3000L UoF Failure Mode Isolation

3.15.3 Class definition

3.15.3.1 FailureModeIsolationCharacteristics

`FailureModeIsolationCharacteristics` is an `<<attributeGroup>>` that collects data which describes the ability to determine that it is the associated failure mode that has occurred.

3.15.3.1.1 Attribute(s)

This `<<attributeGroup>>` has the following attributes:

- `failureModeIsolationAbilityRating`
- `failureModeIsolationAbilityDescription`, zero, one or many
- `failureModeIsolationRate`, zero, one or many

3.15.3.2 FailureModeIsolationItem

`FailureModeIsolationItem` is an `<<extend>>` interface that provides its associated data model to those classes that implement it.

3.15.3.2.1 Class members

Classes that implement the `FailureModeIsolationItem` `<<extend>>` interface are:

- `FailureMode`. Refer to Para 3.14
- `LSAFailureModeGroup`. Refer to Para 3.22

3.15.3.2.2 Associations

The `FailureModeIsolationItem` `<<extend>>` interface has the following associations:

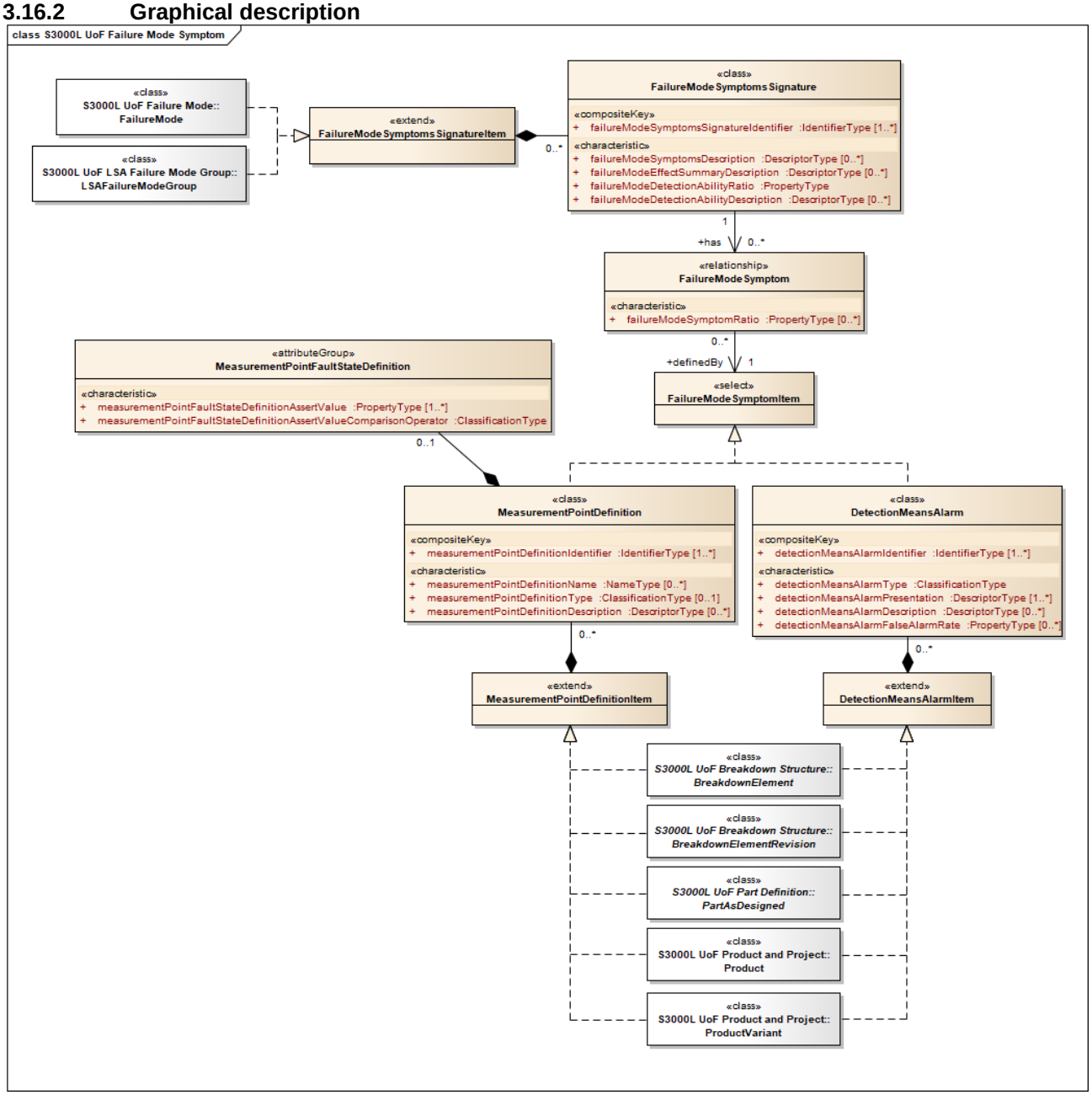
– An aggregate association with zero or one instance of FailureModeIsolationCharacteristics

3.16 S3000L UoF Failure Mode Symptom

3.16.1 Description

The Failure Mode Symptom UoF provides the capability to identify measurable or visible parameters whose appearance can be related, directly or indirectly, to the occurrence of the associated failure mode.

3.16.2 Graphical description



ICN-B6865-S3000L0278-001-01

Fig 20 S3000L UoF Failure Mode Symptom

3.16.3 Class definition

3.16.3.1 DetectionMeansAlarm

DetectionMeansAlarm is a <<class>> that supports the definition and description of an alarm being implemented by a DetectionMeansAlarmItem.

3.16.3.1.1 Attribute(s)

This class has the following attributes:

- detectionMeansAlarmIdentifier, one or many
- detectionMeansAlarmType
- detectionMeansAlarmPresentation, one or many
- detectionMeansAlarmDescription, zero, one or many
- detectionMeansAlarmFalseAlarmRate, zero, one or many

3.16.3.1.2 Implementations

This class implements the following <<extend>> interfaces:

- DigitalFileReferencingItem. Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.16.3.2 DetectionMeansAlarmItem

DetectionMeansAlarmItem is an <<extend>> interface that provides its associated data model to those classes that can have an associated DetectionMeansAlarm.

3.16.3.2.1 Class members

Classes that implement the DetectionMeansAlarmItem <<extend>> interface are:

- BreakdownElement. Refer to Para 3.3
- BreakdownElementRevision. Refer to Para 3.3
- PartAsDesigned. Refer to Para 3.25
- Product. Refer to Para 3.27
- ProductVariant. Refer to Para 3.27

3.16.3.2.2 Associations

The DetectionMeansAlarmItem <<extend>> interface has the following associations:

- An aggregate association with zero, one or many instances of DetectionMeansAlarm

3.16.3.3 FailureModeSymptom

FailureModeSymptom is a <<relationship>> that identifies a measurable or visible parameter whose appearance can be related, directly or indirectly, to the occurrence of the associated failure mode.

3.16.3.3.1 Attribute(s)

This class has the following attributes:

- failureModeSymptomRatio, zero, one or many

3.16.3.3.2 Associations
This class has the following associations:

- A directed definedBy association with one instance of FailureModeSymptomItem

3.16.3.3.3 Implementations
This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- ChangeControlledItem. Refer to Para 3.4
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.16.3.4 FailureModeSymptomItem
FailureModeSymptomItem is a <<select>> interface that identifies items which can provide a measurable or visible parameter whose appearance can be related, directly or indirectly, to the occurrence of the associated failure mode.

3.16.3.4.1 Class members
This <<select>> interface includes the following class members:

- DetectionMeansAlarm
- MeasurementPointDefinition

3.16.3.5 FailureModeSymptomsSignature
FailureModeSymptomsSignature is a <<class>> that identifies a set of failure mode symptoms and effects which can be associated with a failure mode.

Note
Failure mode symptoms can be human observable as well as alarms and measurements.

3.16.3.5.1 Attribute(s)
This class has the following attributes:

- failureModeSymptomsSignatureIdentifier, one or many
- failureModeSymptomsDescription, zero, one or many
- failureModeEffectSummaryDescription, zero, one or many
- failureModeDetectionAbilityRatio
- failureModeDetectionAbilityDescription, zero, one or many

3.16.3.5.2 Associations
This class has the following associations:

- A directed has association with zero, one or many instances of FailureModeSymptom

3.16.3.5.3 Implementations
This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24

- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.16.3.6 FailureModeSymptomsSignatureItem

FailureModeSymptomsSignatureItem is an <<extend>> interface that provides its associated data model to those classes that can have an associated FailureModeSymptomsSignature.

3.16.3.6.1 Class members

Classes that implement the FailureModeSymptomsSignatureItem <<extend>> interface are:

- FailureMode
- LSAFailureModeGroup

3.16.3.6.2 Associations

The FailureModeSymptomsSignatureItem <<extend>> interface has the following associations:

- An aggregate association with zero, one or many instances of FailureModeSymptomsSignature

3.16.3.7 MeasurementPointDefinition

MeasurementPointDefinition is a <<class>> that specifies a characteristic which can be observed for a MeasurementPointDefinitionItem.

3.16.3.7.1 Attribute(s)

This class has the following attributes:

- measurementPointDefinitionIdentifier, one or many
- measurementPointDefinitionName, zero, one or many
- measurementPointDefinitionType, zero or one
- measurementPointDefinitionDescription, zero, one or many

3.16.3.7.2 Associations

This class has the following associations:

- An aggregate association with zero or one instance of MeasurementPointFaultStateDefinition
- An aggregate association with zero or one instance of MeasurementPointDegradedStateDefinition. Refer to Para 3.40

3.16.3.7.3 Implementations

This class implements the following <<extend>> interfaces:

- DigitalFileReferencingItem. Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.16.3.8 MeasurementPointDefinitionItem

MeasurementPointDefinitionItem is an <<extend>> interface that provides its associated data model to those classes that can have an associated MeasurementPointDefinition.

3.16.3.8.1 Class members

Classes that implement the MeasurementPointDefinitionItem <<extend>> interface are:

- BreakdownElement. Refer to Para 3.3
- BreakdownElementRevision. Refer to Para 3.3
- PartAsDesigned. Refer to Para 3.25
- Product. Refer to Para 3.27
- ProductVariant. Refer to Para 3.27

3.16.3.8.2 Associations

The MeasurementPointDefinitionItem <<extend>> interface has the following associations:

- An aggregate association with zero, one or many instances of MeasurementPointDefinition

3.16.3.9 MeasurementPointFaultStateDefinition

MeasurementPointFaultStateDefinition is an <<attributeGroup>> that specifies a measurable condition which, when met, can identify that a failure has occurred.

3.16.3.9.1 Attribute(s)

This <<attributeGroup>> has the following attributes:

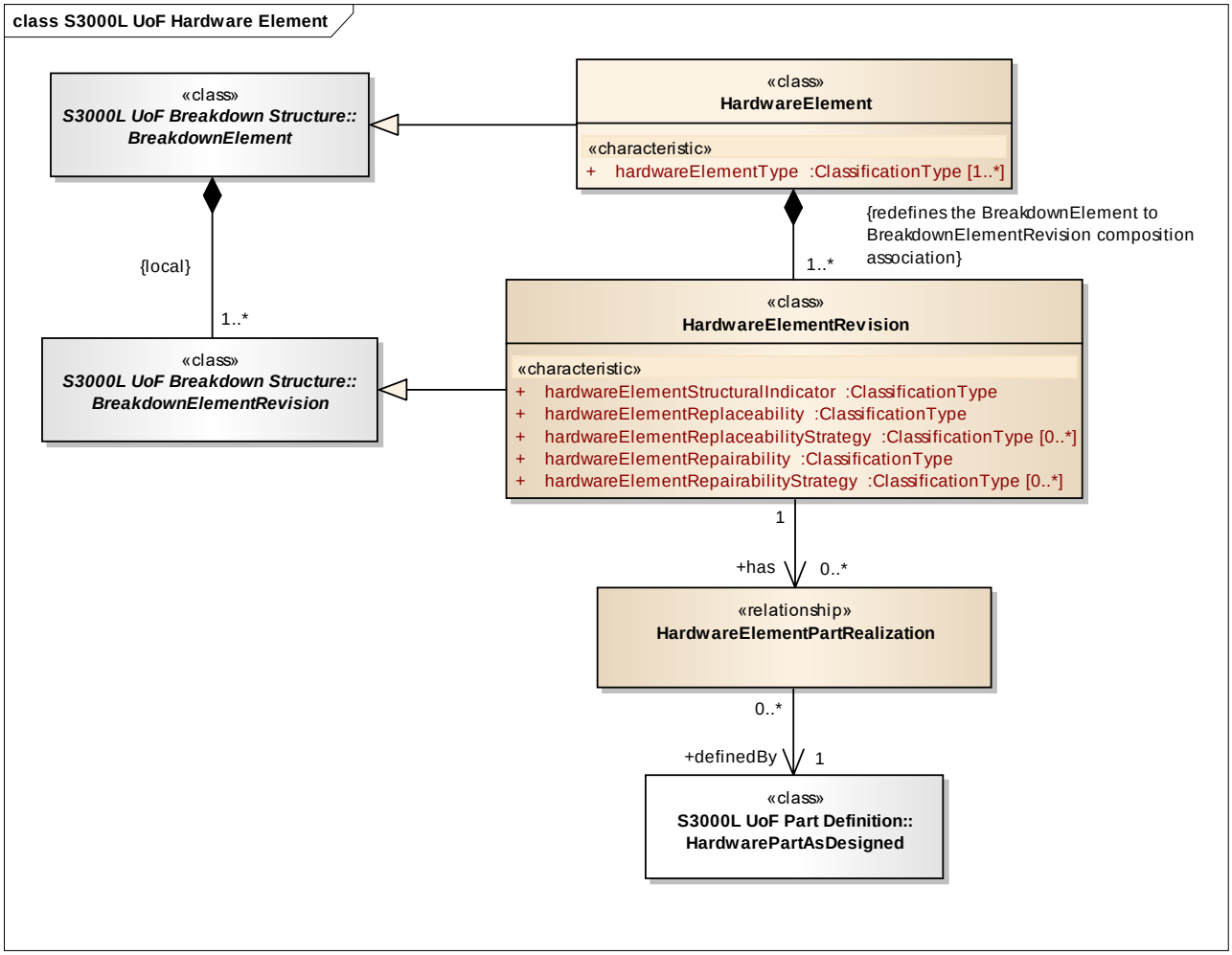
- measurementPointFaultStateDefinitionAssertValue, one or many
- measurementPointFaultStateDefinitionAssertValueComparisonOperator

3.17 S3000L UoF Hardware Element

3.17.1 Description

The Hardware Element UoF provides the capability to specify that an element within a breakdown is hardware and can be associated with the hardware part(s) that fulfill the requirement.

3.17.2 Graphical description



ICN-B6865-S3000L0273-001-01

Fig 21 S3000L UoF Hardware Element

3.17.3 Class definition

3.17.3.1 HardwareElement

`HardwareElement` is a `BreakdownElement` (refer to Para 3.3) that is realized as a `HardwarePartAsDesigned` (refer to Para 3.25).

3.17.3.1.1 Attribute(s)

This class has the following attributes:

- `breakdownElementIdentifier` (inherited from `BreakdownElement`), one or many
- `breakdownElementName` (inherited from `BreakdownElement`), zero, one or many
- `breakdownElementEssentiality` (inherited from `BreakdownElement`), zero or one
- `hardwareElementType`, one or many

3.17.3.1.2 Associations

This class has the following associations:

- An aggregate association with one or many instances of HardwareElementRevision

3.17.3.1.3 Implementations

This class implements the following <<extend>> interfaces:

- AnalysisCandidateItem (inherited from BreakdownElement). Refer to Para 3.21
- BreakdownElementInZoneItem (inherited from BreakdownElement). Refer to Para 3.41
- DecisionTreeAnalysisItem (inherited from BreakdownElement). Refer to Para 3.8
- DetectionMeansAlarmItem (inherited from BreakdownElement). Refer to Para 3.16
- DigitalFileReferencingItem (inherited from BreakdownElement). Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- FailureModeAnalysisItem (inherited from BreakdownElement). Refer to Para 3.14
- InServiceOptimizationAnalysisItem (inherited from BreakdownElement). Refer to Para 3.18
- MeasurementPointDefinitionItem (inherited from BreakdownElement). Refer to Para 3.16
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- PerformanceParameterItem (inherited from BreakdownElement). Refer to Para 3.26
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem (inherited from BreakdownElement). Refer to Para 3.33

3.17.3.2 HardwareElementPartRealization

HardwareElementPartRealization is a <<relationship>> where a HardwareElementRevision relates to an instance of HardwarePartAsDesigned which fulfills the HardwareElement specification.

3.17.3.2.1 Associations

This class has the following associations:

- A directed association with one instance of HardwarePartAsDesigned. Refer to Para 3.25

3.17.3.2.2 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- ChangeControlledItem. Refer to Para 3.4
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- EffectiveOnProductConfigurationItem. Refer to Para 3.28
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24

- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- UsableOnItem. Refer to Para 3.28

3.17.3.3 HardwareElementRevision

HardwareElementRevision is a BreakdownElementRevision (refer to Para 3.3) representing an iteration applied to a HardwareElement.

3.17.3.3.1 Attribute(s)

This class has the following attributes:

- breakdownElementRevisionIdentifier (inherited from BreakdownElementRevision)
- breakdownElementDescription (inherited from BreakdownElementRevision), zero, one or many
- maintenanceSignificantOrRelevant (inherited from BreakdownElementRevision)
- breakdownElementRevisionRationale (inherited from BreakdownElementRevision), zero, one or many
- breakdownElementRevisionDate (inherited from BreakdownElementRevision), zero or one
- breakdownElementRevisionStatus (inherited from BreakdownElementRevision), zero or one
- hardwareElementStructuralIndicator
- hardwareElementReplaceability
- hardwareElementReplaceabilityStrategy, zero, one or many
- hardwareElementRepairability
- hardwareElementRepairabilityStrategy, zero, one or many

3.17.3.3.2 Associations

This class has the following associations:

- A directed has association with zero, one or many instances of BreakdownElementRevisionRelationship (inherited from BreakdownElementRevision). Refer to Para 3.3
- A directed has association with zero, one or many instances of HardwareElementPartRealization

3.17.3.3.3 Implementations

This class implements the following <<extend>> interfaces:

- AnalysisCandidateItem (inherited from BreakdownElementRevision). Refer to Para 3.21
- BreakdownElementInZoneItem (inherited from BreakdownElementRevision). Refer to Para 3.41
- ChangeControlledItem (inherited from BreakdownElementRevision). Refer to Para 3.4
- DetectionMeansAlarmItem (inherited from BreakdownElementRevision). Refer to Para 3.16
- DigitalFileReferencingItem (inherited from BreakdownElementRevision). Refer to Para 3.10

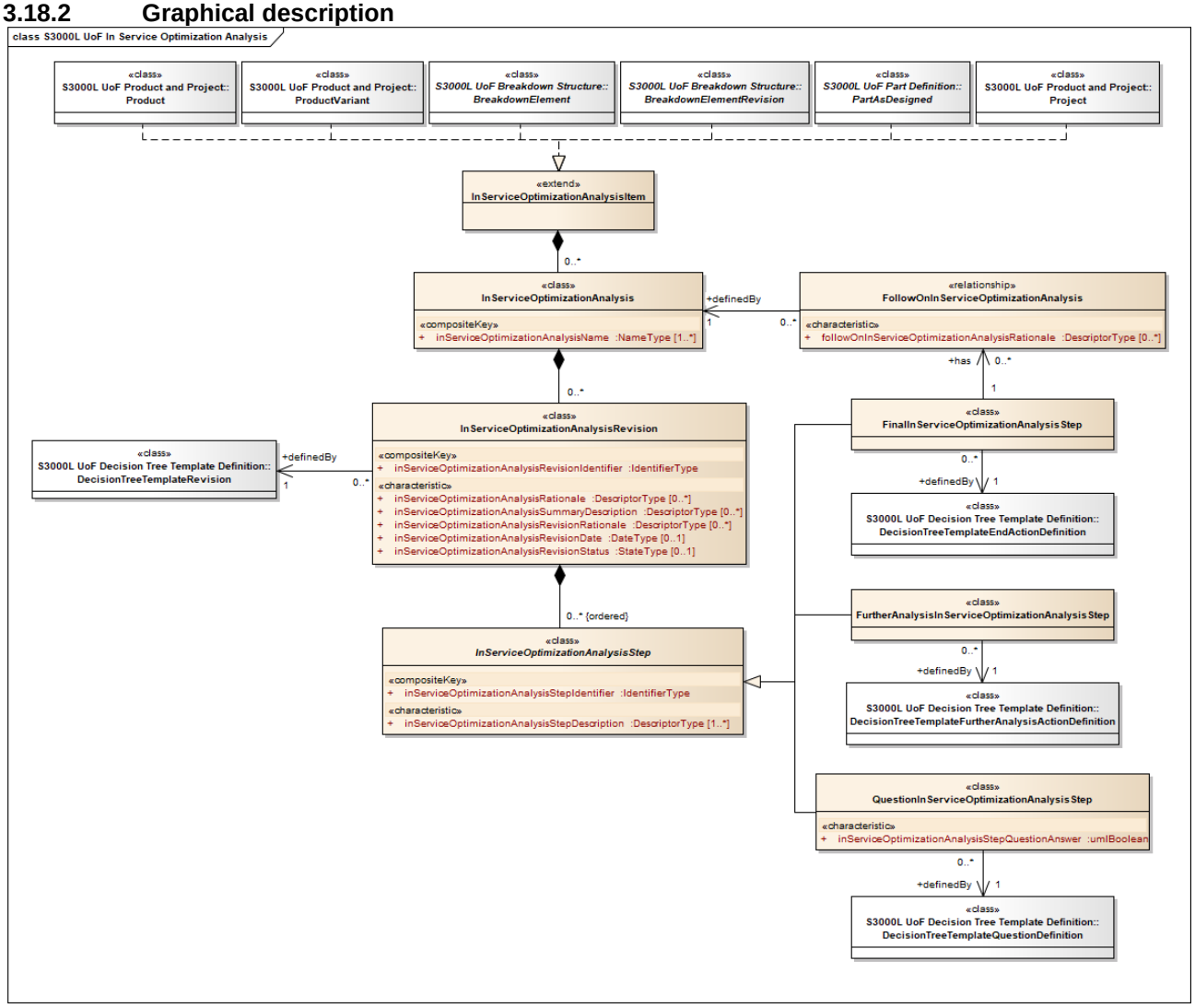
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- FailureModeAnalysisItem (inherited from BreakdownElementRevision). Refer to Para 3.14
- InServiceOptimizationAnalysisItem (inherited from BreakdownElementRevision). Refer to Para 3.18
- MeasurementPointDefinitionItem (inherited from BreakdownElementRevision). Refer to Para 3.16
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- PerformanceParameterItem (inherited from BreakdownElementRevision). Refer to Para 3.26
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- TaskAnalysisItem (inherited from BreakdownElementRevision). Refer to Para 3.39
- TaskRequirementAnalysisItem (inherited from BreakdownElementRevision). Refer to Para 3.37

3.18 S3000L UoF In Service Optimization Analysis

3.18.1 Description

The In Service Optimization Analysis UoF provides the capability to represent the result from an actual analysis carried out for the analyzed item and in accordance with a defined decision tree template.

3.18.2 Graphical description



ICN-B6865-S3000L0290-001-01

Fig 22 S3000L UoF In Service Optimization Analysis

3.18.3 Class definition

3.18.3.1 FinalInServiceOptimizationAnalysisStep

`FinalInServiceOptimizationAnalysisStep` is an `InServiceOptimizationAnalysisStep` that specifies final actions and measures taken together with the decision process conclusion and recommendations.

3.18.3.1.1 Attribute(s)

This class has the following attributes:

- `inServiceOptimizationAnalysisStepIdentifier` (inherited from `InServiceOptimizationAnalysisStep`)
- `inServiceOptimizationAnalysisStepDescription` (inherited from `InServiceOptimizationAnalysisStep`), one or many

3.18.3.1.2 Associations

This class has the following associations:

- A directed has association with zero, one or many instances of FollowOnInServiceOptimizationAnalysis
- A directed definedBy association with one instance of DecisionTreeTemplateEndActionDefinition. Refer to Para 3.8

3.18.3.1.3 Implementations

This class implements the following <<extend>> interfaces:

- DigitalFileReferencingItem (inherited from InServiceOptimizationAnalysisStep). Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.18.3.2 FollowOnInServiceOptimizationAnalysis

FollowOnInServiceOptimizationAnalysis is a <<relationship>> where the outcome from the InServiceOptimizationAnalysis resulted in an additional InServiceOptimizationAnalysis.

3.18.3.2.1 Attribute(s)

This class has the following attributes:

- followOnInServiceOptimizationAnalysisRationale, zero, one or many

3.18.3.2.2 Associations

This class has the following associations:

- A directed definedBy association with one instance of InServiceOptimizationAnalysis

3.18.3.2.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.18.3.3 FurtherAnalysisInServiceOptimizationAnalysisStep

FurtherAnalysisInServiceOptimizationAnalysisStep is an InServiceOptimizationAnalysisStep that specifies actions and measures taken before continuing to the next step defined in the associated DecisionTreeTemplate (refer to Para 3.8).

3.18.3.3.1 Attribute(s)

This class has the following attributes:

- inServiceOptimizationAnalysisStepIdentifier (inherited from InServiceOptimizationAnalysisStep)

- inServiceOptimizationAnalysisStepDescription (inherited from InServiceOptimizationAnalysisStep), one or many

3.18.3.3.2 Associations

This class has the following associations:

- A directed association with one instance of DecisionTreeTemplateFurtherAnalysisActionDefinition. Refer to Para 3.8

3.18.3.3.3 Implementations

This class implements the following <<extend>> interfaces:

- DigitalFileReferencingItem (inherited from InServiceOptimizationAnalysisStep). Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.18.3.4 InServiceOptimizationAnalysis

InServiceOptimizationAnalysis is a <<class>> that represents the result from an in-service optimization analysis carried out for the InServiceOptimizationAnalysisItem.

3.18.3.4.1 Attribute(s)

This class has the following attributes:

- inServiceOptimizationAnalysisName, one or many

3.18.3.4.2 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of InServiceOptimizationAnalysisRevision

3.18.3.4.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.18.3.5 InServiceOptimizationAnalysisItem

InServiceOptimizationAnalysisItem is an <<extend>> interface that provides its associated data model to those classes that can have an associated InServiceOptimizationAnalysis.

3.18.3.5.1 Class members

Classes that implement the InServiceOptimizationAnalysisItem <<extend>> interface are:

- BreakdownElement. Refer to Para 3.3
- BreakdownElementRevision. Refer to Para 3.3
- PartAsDesigned. Refer to Para 3.25
- Product. Refer to Para 3.27
- ProductVariant. Refer to Para 3.27
- Project. Refer to Para 3.27

3.18.3.5.2 Associations

The InServiceOptimizationAnalysisItem <<extend>> interface has the following associations:

- An aggregate association with zero, one or many instances of InServiceOptimizationAnalysis

3.18.3.6 InServiceOptimizationAnalysisRevision

InServiceOptimizationAnalysisRevision is a <<class>> representing an iteration applied to an InServiceOptimizationAnalysis.

3.18.3.6.1 Attribute(s)

This class has the following attributes:

- inServiceOptimizationAnalysisRevisionIdentifier
- inServiceOptimizationAnalysisRationale, zero, one or many
- inServiceOptimizationAnalysisSummaryDescription, zero, one or many
- inServiceOptimizationAnalysisRevisionRationale, zero, one or many
- inServiceOptimizationAnalysisRevisionDate, zero or one
- inServiceOptimizationAnalysisRevisionStatus, zero or one

3.18.3.6.2 Associations

This class has the following associations:

- An ordered aggregate association with zero, one or many instances of InServiceOptimizationAnalysisStep
- A directed definedBy association with one instance of DecisionTreeTemplateRevision. Refer to Para 3.8

3.18.3.6.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.18.3.7 InServiceOptimizationAnalysisStep

InServiceOptimizationAnalysisStep is a <<class>> that represents the results from an individual step in the associated decision process.

3.18.3.7.1 Attribute(s)

This class has the following attributes:

- inServiceOptimizationAnalysisStepIdentifier
- inServiceOptimizationAnalysisStepDescription, one or many

3.18.3.7.2 Implementations

This class implements the following <<extend>> interfaces:

- DigitalFileReferencingItem. Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.18.3.8 QuestionInServiceOptimizationAnalysisStep

QuestionInServiceOptimizationAnalysisStep is an InServiceOptimizationAnalysisStep that specifies the answer to a question defined in the associated DecisionTreeTemplate (refer to Para 3.8).

3.18.3.8.1 Attribute(s)

This class has the following attributes:

- inServiceOptimizationAnalysisStepIdentifier (inherited from InServiceOptimizationAnalysisStep)
- inServiceOptimizationAnalysisStepDescription (inherited from InServiceOptimizationAnalysisStep), one or many
- inServiceOptimizationAnalysisStepQuestionAnswer

3.18.3.8.2 Associations

This class has the following associations:

- A directed definedBy association with one instance of DecisionTreeTemplateQuestionDefinition. Refer to Para 3.8

3.18.3.8.3 Implementations

This class implements the following <<extend>> interfaces:

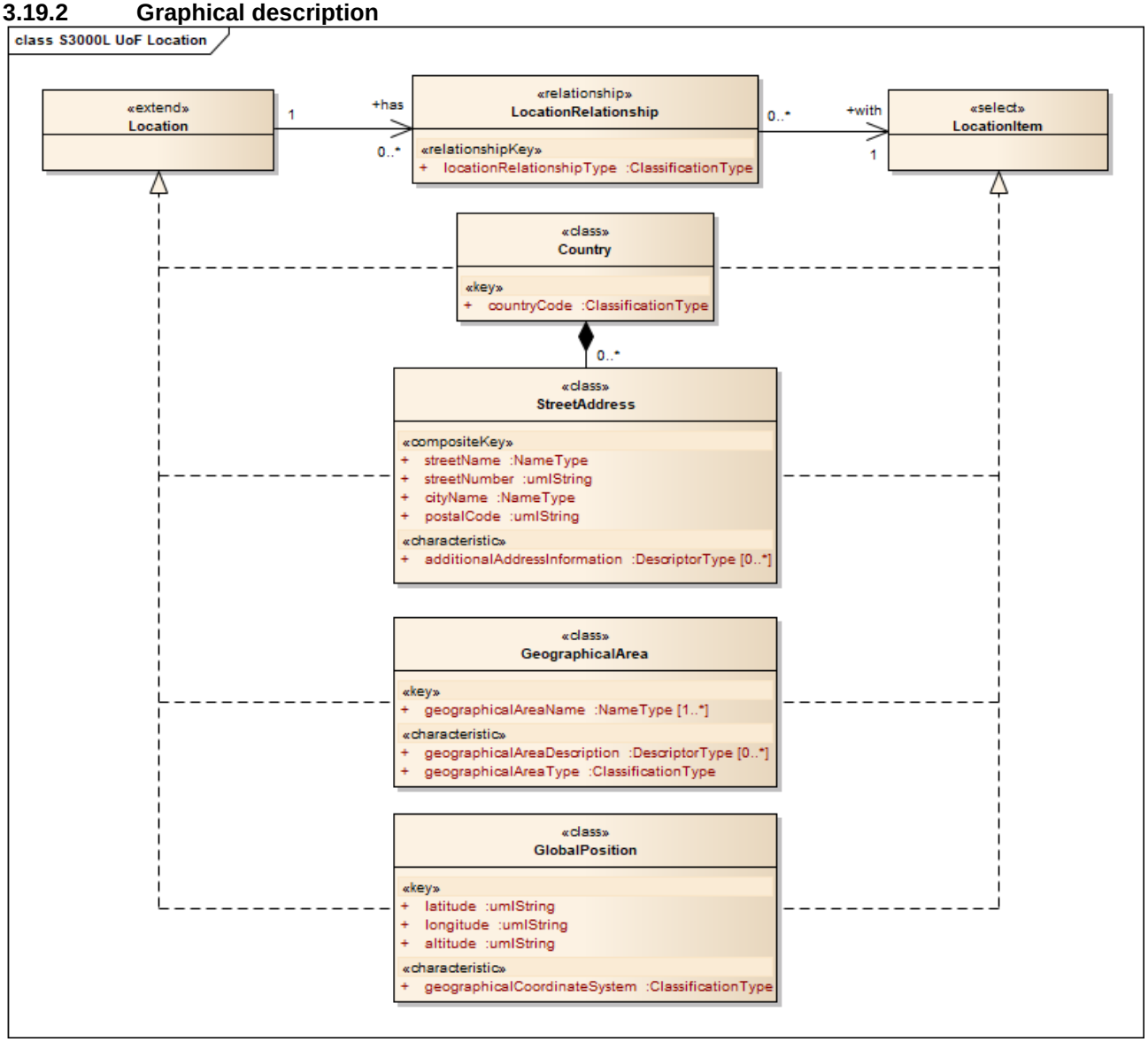
- DigitalFileReferencingItem (inherited from InServiceOptimizationAnalysisStep). Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.19 S3000L UoF Location

3.19.1 Description

The Location UoF provides the capability to define a geographic location.

3.19.2 Graphical description



ICN-B6865-S3000L0271-001-01

Fig 23 S3000L UoF Location

3.19.3 Class definition

3.19.3.1 Country

`Country` is a self-governing political entity, occupying a particular territory.

3.19.3.1.1 Attribute(s)

This class has the following attributes:

- `countryCode`

3.19.3.1.2 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of `StreetAddress`

3.19.3.1.3 Implementations

This class implements the following `<<extend>>` interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- EnvironmentDefinitionItem. Refer to Para 3.12
- Location. Refer to Para 3.19
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.19.3.2 GeographicalArea

GeographicalArea is a <<class>> that represents a particular extent of space.

3.19.3.2.1 Attribute(s)

This class has the following attributes:

- geographicalAreaName, one or many
- geographicalAreaDescription, zero, one or many
- geographicalAreaType

3.19.3.2.2 Implementations

This class implements the following <<extend>> interfaces:

- DigitalFileReferencingItem. Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- EnvironmentDefinitionItem. Refer to Para 3.12
- Location. Refer to Para 3.19
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.19.3.3 GlobalPosition

GlobalPosition is a <<class>> that identifies a point in space by a set of coordinates.

3.19.3.3.1 Attribute(s)

This class has the following attributes:

- latitude
- longitude
- altitude
- geographicalCoordinateSystem

3.19.3.3.2 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- EnvironmentDefinitionItem. Refer to Para 3.12
- Location. Refer to Para 3.19
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D

- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.19.3.4 Location

Location is an <<extend>> interface that provides its associated data model to those classes that implement it.

3.19.3.4.1 Class members

Classes that implement the Location <<extend>> interface are:

- Country
- GeographicalArea
- GlobalPosition
- StreetAddress

3.19.3.4.2 Associations

The Location <<extend>> interface has the following associations:

- A directed has association with zero, one or many instances of LocationRelationship

3.19.3.5 LocationItem

LocationItem is a <<select>> interface that identifies items which can be selected to provide the definition of a geographic location.

3.19.3.5.1 Class members

This <<select>> interface includes the following class members:

- Country
- GeographicalArea
- GlobalPosition
- StreetAddress

3.19.3.6 LocationRelationship

LocationRelationship is a <<relationship>> where one LocationItem relates to another LocationItem

3.19.3.6.1 Attribute(s)

This class has the following attributes:

- locationRelationshipType

3.19.3.6.2 Associations

This class has the following associations:

- A directed with association with zero, one or many instances of LocationItem

3.19.3.6.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.19.3.7 StreetAddress

`StreetAddress` is a <<class> that represents a locatable position along a road.

3.19.3.7.1 Attribute(s)

This class has the following attributes:

- `streetName`
- `streetNumber`
- `cityName`
- `postalCode`
- `additionalAddressInformation`, zero, one or many

3.19.3.7.2 Implementations

This class implements the following `<<extend>>` interfaces:

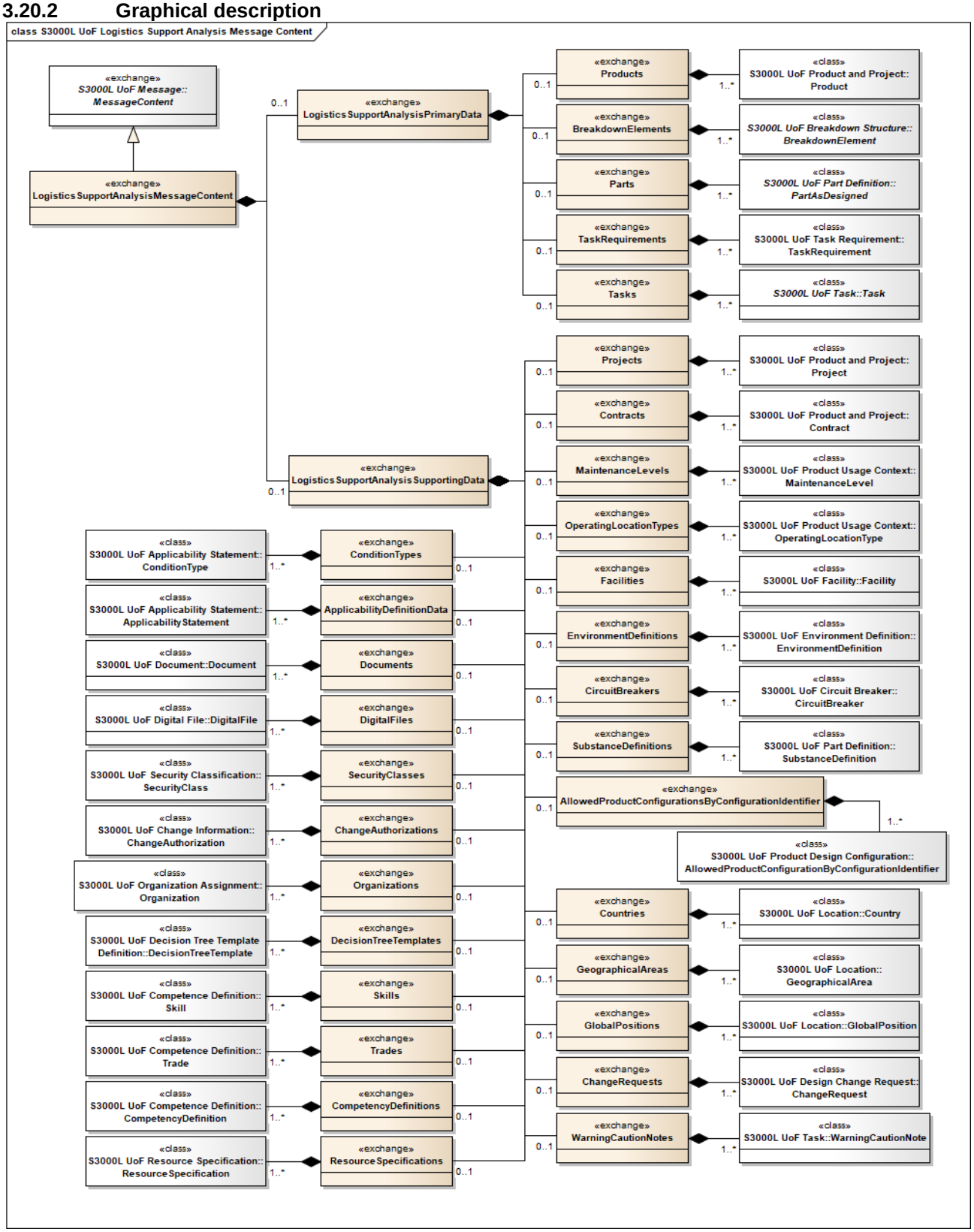
- `DigitalFileReferencingItem`. Refer to Para 3.10
- `DocumentReferencingItem` (inherited from `BaseObject`). Refer to Para 3.11
- `EnvironmentDefinitionItem`. Refer to Para 3.12
- `Location`. Refer to Para 3.19
- `OrganizationReferencingItem` (inherited from `BaseObject`). Refer to Para 3.24
- `ProjectSpecificExtensionItem` (inherited from `BaseObject`). Refer to SX002D
- `RemarkItem` (inherited from `BaseObject`). Refer to Para 3.31

3.20 S3000L UoF Logistics Support Analysis Message Content

3.20.1 Description

The Logistics Support Analysis Message Content UoF defines the collection of information that can be exchanged for Logistics Support Analysis in the context of the S-Series IPS specifications.

3.20.2 Graphical description



ICN-B6865-S3000L0296-001-01

Fig 24 S3000L UoF Logistics Support Analysis Message Content

3.20.3 Class definition

3.20.3.1 AllowedProductConfigurationsByConfigurationIdentifier

AllowedProductConfigurationsByConfigurationIdentifier is a wrapper element that contains all instances of AllowedProductConfigurationByConfigurationIdentifier that are in scope for a data exchange.

3.20.3.1.1 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of AllowedProductConfigurationByConfigurationIdentifier. Refer to Para 3.28

3.20.3.2 ApplicabilityDefinitionData

ApplicabilityDefinitionData is a wrapper element that contains all instances of ApplicabilityStatement that are in scope for a data exchange.

3.20.3.2.1 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of ApplicabilityStatement. Refer to Para 3.2

3.20.3.3 BreakdownElements

BreakdownElements is a wrapper element that contains all instances of BreakdownElement that are in scope for a data exchange.

3.20.3.3.1 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of BreakdownElement. Refer to Para 3.3

3.20.3.4 ChangeAuthorizations

ChangeAuthorizations is a wrapper element that contains all instances of ChangeAuthorization that are in scope for a data exchange.

3.20.3.4.1 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of ChangeAuthorization. Refer to Para 3.4

3.20.3.5 ChangeRequests

ChangeRequests is a wrapper element that contains all instances of ChangeRequest that are in scope for a data exchange.

3.20.3.5.1 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of ChangeRequest. Refer to Para 3.9

3.20.3.6 CircuitBreakers

CircuitBreakers is a wrapper element that contains all instances of CircuitBreaker that are in scope for a data exchange.

3.20.3.6.1 Associations
This class has the following associations:

– An aggregate association with zero, one or many instances of CircuitBreaker. Refer to Para 3.5

3.20.3.7 CompetencyDefinitions
CompetencyDefinitions is a wrapper element that contains all instances of CompetencyDefinition that are in scope for a data exchange.

3.20.3.7.1 Associations
This class has the following associations:

– An aggregate association with zero, one or many instances of CompetencyDefinition. Refer to Para 3.6

3.20.3.8 ConditionTypes
ConditionTypes is a wrapper element that contains all instances of ConditionType that are in scope for a data exchange.

3.20.3.8.1 Associations
This class has the following associations:

– An aggregate association with zero, one or many instances of ConditionType. Refer to Para 3.2

3.20.3.9 Contracts
Contracts is a wrapper element that contains all instances of Contract that are in scope for a data exchange.

3.20.3.9.1 Associations
This class has the following associations:

– An aggregate association with zero, one or many instances of Contract. Refer to Para 3.27

3.20.3.10 Countries
Countries is a wrapper element that contains all instances of Country that are in scope for a data exchange.

3.20.3.10.1 Associations
This class has the following associations:

– An aggregate association with zero, one or many instances of Country. Refer to Para 3.19

3.20.3.11 DecisionTreeTemplates
DecisionTreeTemplates is a wrapper element that contains all instances of DecisionTreeTemplate that are in scope for a data exchange.

3.20.3.11.1 Associations
This class has the following associations:

– An aggregate association with zero, one or many instances of DecisionTreeTemplate. Refer to Para 3.8

3.20.3.12 DigitalFiles
DigitalFiles is a wrapper element that contains all instances of DigitalFile that are in scope for a data exchange.

3.20.3.12.1 Associations
This class has the following associations:

– An aggregate association with zero, one or many instances of DigitalFile. Refer to Para 3.10

3.20.3.13 Documents
Documents is a wrapper element that contains all instances of Document that are in scope for a data exchange.

3.20.3.13.1 Associations
This class has the following associations:

– An aggregate association with zero, one or many instances of Document. Refer to Para 3.11

3.20.3.14 EnvironmentDefinitions
EnvironmentDefinitions is a wrapper element that contains all instances of EnvironmentDefinition that are in scope for a data exchange.

3.20.3.14.1 Associations
This class has the following associations:

– An aggregate association with zero, one or many instances of EnvironmentDefinition. Refer to Para 3.12

3.20.3.15 Facilities
Facilities is a wrapper element that contains all instances of Facility that are in scope for a data exchange.

3.20.3.15.1 Associations
This class has the following associations:

– An aggregate association with zero, one or many instances of Facility. Refer to Para 3.13

3.20.3.16 GeographicalAreas
GeographicalAreas is a wrapper element that contains all instances of GeographicalArea that are in scope for a data exchange.

3.20.3.16.1 Associations
This class has the following associations:

– An aggregate association with zero, one or many instances of GeographicalArea. Refer to Para 3.19

3.20.3.17 GlobalPositions
GlobalPositions is a wrapper element that contains all instances of GlobalPosition that are in scope for a data exchange.

3.20.3.17.1 Associations
This class has the following associations:

– An aggregate association with zero, one or many instances of GlobalPosition. Refer to Para 3.19

3.20.3.18 LogisticsSupportAnalysisMessageContent
LogisticsSupportAnalysisMessageContent is a MessageContent (refer to Para 3.23) that contains data resulting from Logistics Support Analysis activities.

3.20.3.18.1 Associations
This class has the following associations:

- An aggregate association with zero or one instance of LogisticsSupportAnalysisPrimaryData
- An aggregate association with zero or one instance of LogisticsSupportAnalysisSupportingData

3.20.3.19 LogisticsSupportAnalysisPrimaryData
LogisticsSupportAnalysisPrimaryData is a subset of the LogisticsSupportAnalysisMessageContent <<exchange>> dataset that include the primary data used in, and resulting from, the Logistics Support Analysis activities.

3.20.3.19.1 Associations
This class has the following associations:

- An aggregate association with zero or one instance of BreakdownElements
- An aggregate association with zero or one instance of Parts
- An aggregate association with zero or one instance of Products
- An aggregate association with zero or one instance of TaskRequirements
- An aggregate association with zero or one instance of Tasks

3.20.3.20 LogisticsSupportAnalysisSupportingData
LogisticsSupportAnalysisSupportingData is a subset of the LogisticsSupportAnalysisMessageContent <<exchange>> dataset that include the data that support the Logistics Support Analysis activities.

3.20.3.20.1 Associations
This class has the following associations:

- An aggregate association with zero or one instance of AllowedProductConfigurationsByConfigurationIdentifier
- An aggregate association with zero or one instance of ApplicabilityDefinitionData
- An aggregate association with zero or one instance of ChangeAuthorizations
- An aggregate association with zero or one instance of ChangeRequests
- An aggregate association with zero or one instance of CircuitBreakers
- An aggregate association with zero or one instance of CompetencyDefinitions
- An aggregate association with zero or one instance of ConditionTypes
- An aggregate association with zero or one instance of Contracts
- An aggregate association with zero or one instance of Countries
- An aggregate association with zero or one instance of DecisionTreeTemplates
- An aggregate association with zero or one instance of DigitalFiles
- An aggregate association with zero or one instance of Documents
- An aggregate association with zero or one instance of EnvironmentDefinitions
- An aggregate association with zero or one instance of Facilities
- An aggregate association with zero or one instance of GeographicalAreas
- An aggregate association with zero or one instance of GlobalPositions
- An aggregate association with zero or one instance of MaintenanceLevels
- An aggregate association with zero or one instance of OperatingLocationTypes
- An aggregate association with zero or one instance of Organizations
- An aggregate association with zero or one instance of Projects
- An aggregate association with zero or one instance of ResourceSpecifications

- An aggregate association with zero or one instance of SecurityClasses
- An aggregate association with zero or one instance of Skills
- An aggregate association with zero or one instance of SubstanceDefinitions
- An aggregate association with zero or one instance of Trades
- An aggregate association with zero or one instance of WarningCautionNotes

3.20.3.21 MaintenanceLevels

MaintenanceLevels is a wrapper element that contains all instances of MaintenanceLevel that are in scope for a data exchange.

3.20.3.21.1 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of MaintenanceLevel. Refer to Para 3.29

3.20.3.22 OperatingLocationTypes

OperatingLocationTypes is a wrapper element that contains all instances of OperatingLocationType that are in scope for a data exchange.

3.20.3.22.1 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of OperatingLocationType. Refer to Para 3.29

3.20.3.23 Organizations

Organizations is a wrapper element that contains all instances of Organization that are in scope for a data exchange.

3.20.3.23.1 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of Organization. Refer to Para 3.24

3.20.3.24 Parts

Parts is a wrapper element that contains all instances of PartAsDesigned that are in scope for a data exchange.

3.20.3.24.1 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of PartAsDesigned. Refer to Para 3.25

3.20.3.25 Products

Products is a wrapper element that contains all instances of Product that are in scope for a data exchange.

3.20.3.25.1 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of Product. Refer to Para 3.27

3.20.3.26 Projects

Projects is a wrapper element that contains all instances of Project that are in scope for a data exchange.

3.20.3.26.1 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of Project. Refer to Para 3.27

3.20.3.27 ResourceSpecifications

ResourceSpecifications is a wrapper element that contains all instances of ResourceSpecification that are in scope for a data exchange.

3.20.3.27.1 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of ResourceSpecification. Refer to Para 3.38

3.20.3.28 SecurityClasses

SecurityClasses is a wrapper element that contains all instances of SecurityClass that are in scope for a data exchange.

3.20.3.28.1 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of SecurityClass. Refer to Para 3.33

3.20.3.29 Skills

Skills is a wrapper element that contains all instances of Skill that are in scope for a data exchange.

3.20.3.29.1 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of Skill. Refer to Para 3.6

3.20.3.30 SubstanceDefinitions

SubstanceDefinitions is a wrapper element that contains all instances of SubstanceDefinition that are in scope for a data exchange.

3.20.3.30.1 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of SubstanceDefinition. Refer to Para 3.25

3.20.3.31 TaskRequirements

TaskRequirements is a wrapper element that contains all instances of TaskRequirement that are in scope for a data exchange.

3.20.3.31.1 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of TaskRequirement. Refer to Para 3.37

3.20.3.32 Tasks

`Tasks` is a wrapper element that contains all instances of `Task` that are scope for a data exchange.

3.20.3.32.1 Associations

This class has the following associations:

– An aggregate association with zero, one or many instances of `Task`. Refer to Para 3.36

3.20.3.33 Trades

`Trades` is a wrapper element that contains all instances of `Trade` that are in scope for a data exchange.

3.20.3.33.1 Associations

This class has the following associations:

– An aggregate association with zero, one or many instances of `Trade`. Refer to Para 3.6

3.20.3.34 WarningCautionNotes

`WarningCautionNotes` is a wrapper element that contains all instances of `WarningCautionNote` that are in scope for a data exchange.

3.20.3.34.1 Associations

This class has the following associations:

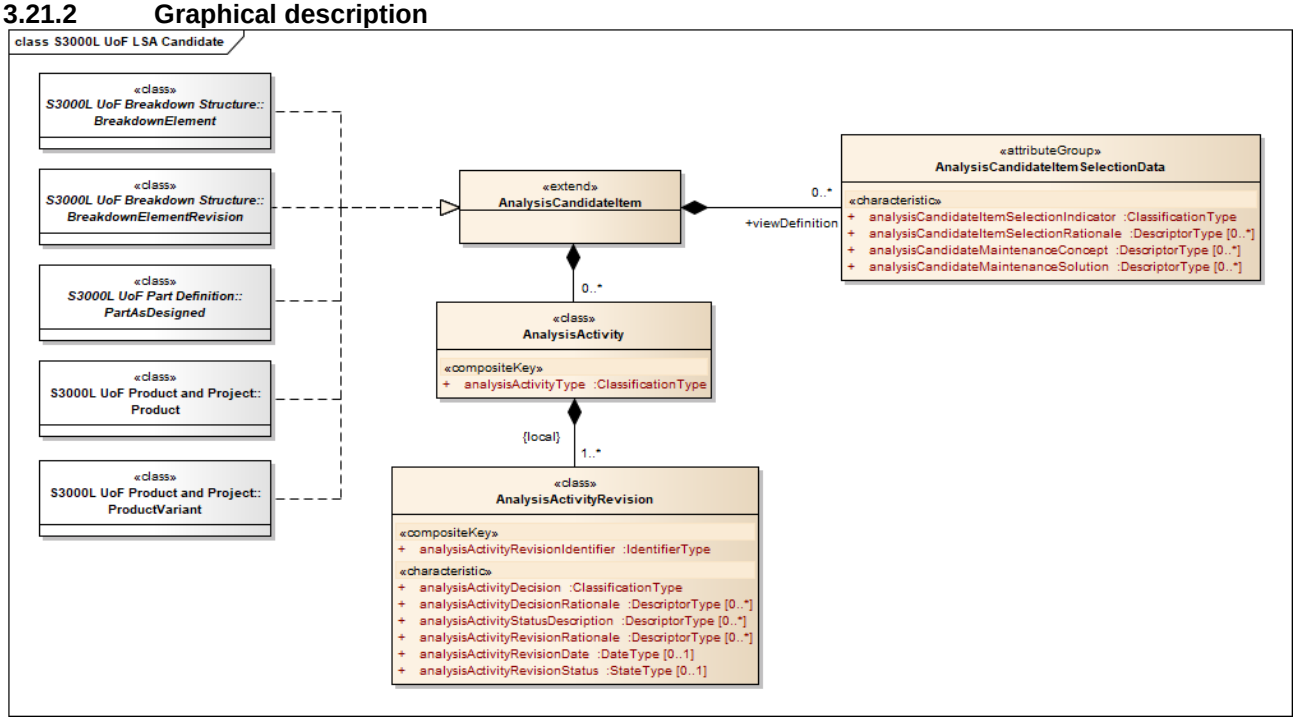
– An aggregate association with zero, one or many instances of `WarningCautionNote`. Refer to Para 3.36

3.21 S3000L UoF LSA Candidate

3.21.1 Description

The LSA Candidate UoF provides the capability to define decisions and results associated with Logistics Support Analysis (LSA) activities.

3.21.2 Graphical description



ICN-B6865-S3000L0223-004-01

Fig 25 S3000L UoF LSA Candidate

3.21.3 Class definition

3.21.3.1 AnalysisActivity

`AnalysisActivity` is a `<<class>>` that represents the objective for, and outcome of, an analysis carried out for the `AnalysisCandidateItem`.

3.21.3.1.1 Attribute(s)

This class has the following attributes:

- `analysisActivityType`

3.21.3.1.2 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of `AnalysisActivityRevision`

3.21.3.1.3 Implementations

This class implements the following `<<extend>>` interfaces:

- `ApplicabilityStatementItem`. Refer to Para 3.2
- `DocumentReferencingItem` (inherited from `BaseObject`). Refer to Para 3.11
- `OrganizationReferencingItem` (inherited from `BaseObject`). Refer to Para 3.24
- `ProjectSpecificExtensionItem` (inherited from `BaseObject`). Refer to SX002D
- `RemarkItem` (inherited from `BaseObject`). Refer to Para 3.31
- `SecurityClassificationItem`. Refer to Para 3.33

3.21.3.2 AnalysisActivityRevision

AnalysisActivityRevision is a <<class>> representing an iteration applied to an AnalysisActivity.

3.21.3.2.1 Attribute(s)

This class has the following attributes:

- analysisActivityRevisionIdentifier
- analysisActivityDecision
- analysisActivityDecisionRationale, zero, one or many
- analysisActivityStatusDescription, zero, one or many
- analysisActivityRevisionRationale, zero, one or many
- analysisActivityRevisionDate, zero or one
- analysisActivityRevisionStatus, zero or one

3.21.3.2.2 Implementations

This class implements the following <<extend>> interfaces:

- ChangeControlledItem. Refer to Para 3.4
- DigitalFileReferencingItem. Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.21.3.3 AnalysisCandidateItem

AnalysisCandidateItem is an <<extend>> interface that provides its associated data model to those classes that can have an associated AnalysisActivity.

3.21.3.3.1 Class members

Classes that implement the AnalysisCandidateItem <<extend>> interface are:

- BreakdownElement. Refer to Para 3.3
- BreakdownElementRevision. Refer to Para 3.3
- PartAsDesigned. Refer to Para 3.25
- Product. Refer to Para 3.27
- ProductVariant. Refer to Para 3.27

3.21.3.3.2 Associations

The AnalysisCandidateItem <<extend>> interface has the following associations:

- An aggregate association with zero, one or many instances of AnalysisActivity
- An aggregate viewDefinition association with zero, one or many instances of AnalysisCandidateItemSelectionData

3.21.3.4 AnalysisCandidateItemSelectionData

AnalysisCandidateItemSelectionData is an <<attributeGroup>> that summarizes decisions made for the AnalysisCandidateItem from a support analysis activities perspective.

3.21.3.4.1 Attribute(s)

This <<attributeGroup>> has the following attributes:

- analysisCandidateItemSelectionIndicator
- analysisCandidateItemSelectionRationale, zero, one or many
- analysisCandidateMaintenanceConcept, zero, one or many
- analysisCandidateMaintenanceSolution, zero, one or many

3.21.3.4.2 Implementations

This class implements the following <<extend>> interfaces:

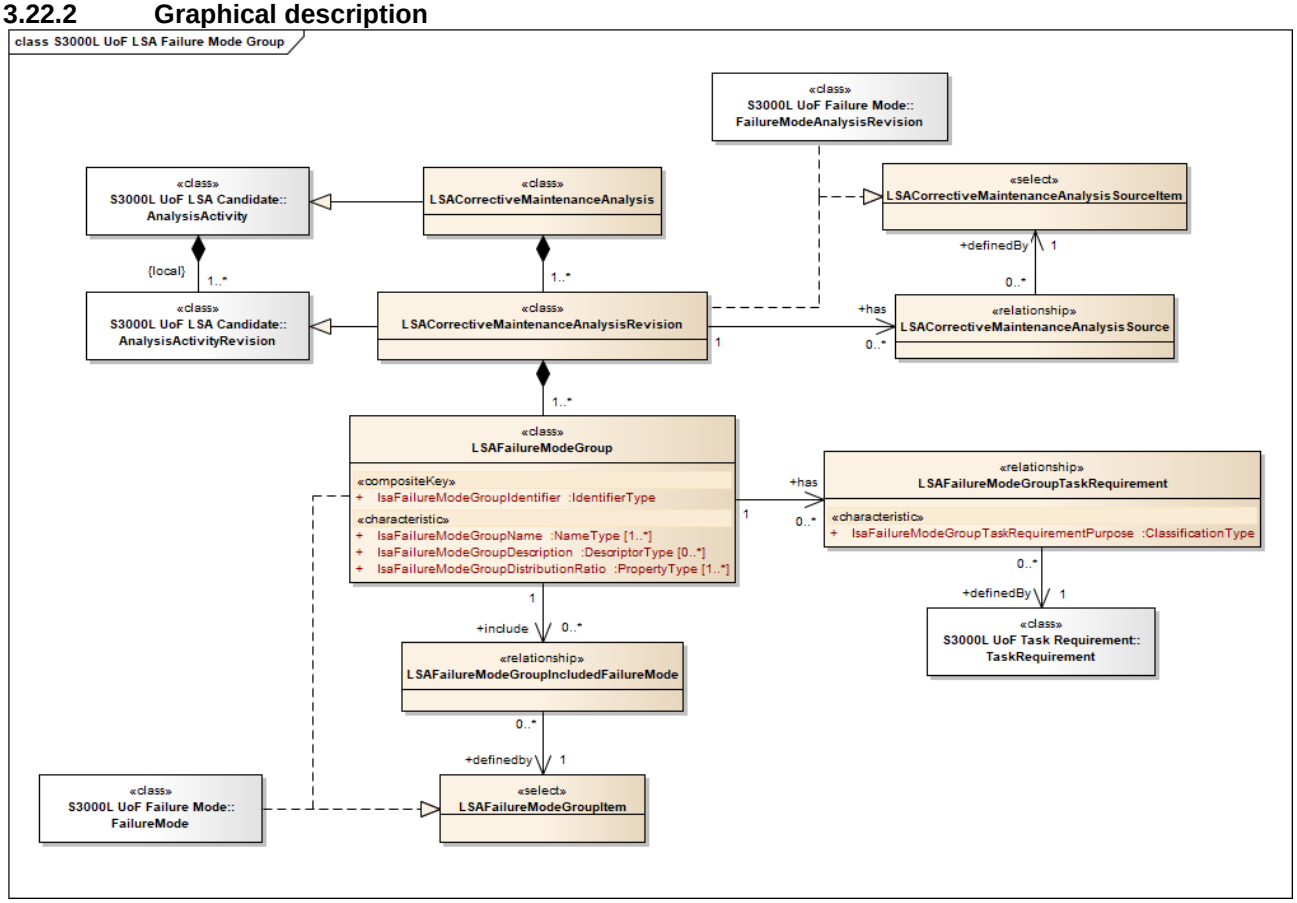
- ApplicabilityStatementItem. Refer to Para 3.2

3.22 S3000L UoF LSA Failure Mode Group

3.22.1 Description

The Failure Mode Group UoF provides the capability to group failure modes that result in the same corrective maintenance activities.

3.22.2 Graphical description



ICN-B6865-S3000L0277-001-01

Fig 26 S3000L UoF LSA Failure Mode Group

3.22.3 Class definition

3.22.3.1 LSACorrectiveMaintenanceAnalysis

LSACorrectiveMaintenanceAnalysis is an AnalysisActivity (refer to Para 3.21) that represents the outcome of corrective maintenance analysis carried out for the AnalysisCandidateItem (refer to Para 3.21).

3.22.3.1.1 Attribute(s)

This class has the following attributes:

- analysisActivityType (inherited from AnalysisActivity)

3.22.3.1.2 Associations

This class has the following associations:

- An aggregate association with one or many instances of LSACorrectiveMaintenanceAnalysisRevision

3.22.3.1.3 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem (inherited from AnalysisActivity). Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem (inherited from AnalysisActivity). Refer to Para 3.33

3.22.3.2 LSACorrectiveMaintenanceAnalysisRevision

LSACorrectiveMaintenanceAnalysisRevision is an AnalysisActivityRevision (refer to Para 3.21) representing an iteration applied to LSACorrectiveMaintenanceAnalysis.

3.22.3.2.1 Attribute(s)

This class has the following attributes:

- analysisActivityRevisionIdentifier (inherited from AnalysisActivityRevision)
- analysisActivityDecision (inherited from AnalysisActivityRevision)
- analysisActivityDecisionRationale (inherited from AnalysisActivityRevision), zero, one or many
- analysisActivityStatusDescription (inherited from AnalysisActivityRevision), zero, one or many
- analysisActivityRevisionRationale (inherited from AnalysisActivityRevision), zero, one or many
- analysisActivityRevisionDate (inherited from AnalysisActivityRevision), zero or one
- analysisActivityRevisionStatus (inherited from AnalysisActivityRevision), zero or one

3.22.3.2.2 Associations

This class has the following associations:

- An aggregate association with one or many instances of LSAFailureModeGroup
- A directed has association with zero, one or many instances of LSACorrectiveMaintenanceAnalysisSource

3.22.3.2.3 Implementations

This class implements the following <<extend>> interfaces:

- ChangeControlledItem (inherited from AnalysisActivityRevision). Refer to Para 3.4
- DigitalFileReferencingItem (inherited from AnalysisActivityRevision). Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.22.3.3 LSACorrectiveMaintenanceAnalysisSource

LSACorrectiveMaintenanceAnalysisSource is a <<relationship>> where an LSACorrectiveMaintenanceAnalysisRevision relates to the failure mode related source information which can be used to identify a required corrective maintenance task.

3.22.3.3.1 Associations

This class has the following associations:

- A directed definedBy association with one instance of LSACorrectiveMaintenanceAnalysisSourceItem

3.22.3.3.2 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.22.3.4 LSACorrectiveMaintenanceAnalysisSourceItem

LSACorrectiveMaintenanceAnalysisSourceItem is a <<select>> interface that identifies the failure mode source information to be used to define the required corrective maintenance task.

3.22.3.4.1 Class members

This <<select>> interface includes the following class members:

- FailureModeAnalysisRevision. Refer to Para 3.14
- LSACorrectiveMaintenanceAnalysisRevision

3.22.3.5 LSAFailureModeGroup

LSAFailureModeGroup is a <<class>> that represents a set of failure modes that leads to the same set of actions for its detection, isolation and rectifying intervention.

3.22.3.5.1 Attribute(s)

This class has the following attributes:

- lsaFailureModeGroupIdentifier
- lsaFailureModeGroupName, one or many
- lsaFailureModeGroupDescription, zero, one or many
- lsaFailureModeGroupDistributionRatio, one or many

3.22.3.5.2 Associations
This class has the following associations:

- A directed include association with zero, one or many instances of LSAFailureModeGroupIncludedFailureMode
- A directed has association with zero, one or many instances of LSAFailureModeGroupTaskRequirement

3.22.3.5.3 Implementations
This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- FailureModeIsolationItem. Refer to Para 3.15
- FailureModeSymptomsSignatureItem. Refer to Para 3.16
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.22.3.6 LSAFailureModeGroupIncludedFailureMode
LSAFailureModeGroupIncludedFailureMode is a <<relationship>> that identifies a FailureMode (refer to Para 3.14) which is a member of the LSAFailureModeGroup.

3.22.3.6.1 Associations
This class has the following associations:

- A directed definedBy association with one instance of LSAFailureModeGroupItem

3.22.3.6.2 Implementations
This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.22.3.7 LSAFailureModeGroupItem
LSAFailureModeGroupItem is a <<select>> interface that identifies failure modes which are included in the LSAFailureModeGroup.

3.22.3.7.1 Class members
This <<select>> interface includes the following class members:

- FailureMode. Refer to Para 3.14
- LSAFailureModeGroup

3.22.3.8 LSAFailureModeGroupTaskRequirement
LSAFailureModeGroupTaskRequirement is a <<relationship>> where an LSAFailureModeGroup relates to the TaskRequirement (refer to Para 3.37)

needed in the process to detect and isolate a failure, and to restore the AnalysisCandidateItem (refer to Para 3.21).

3.22.3.8.1 Attribute(s)

This class has the following attributes:

- lsaFailureModeGroupTaskRequirementPurpose

3.22.3.8.2 Associations

This class has the following associations:

- A directed definedBy association with one instance of TaskRequirement. Refer to Para 3.37

3.22.3.8.3 Implementations

This class implements the following <<extend>> interfaces:

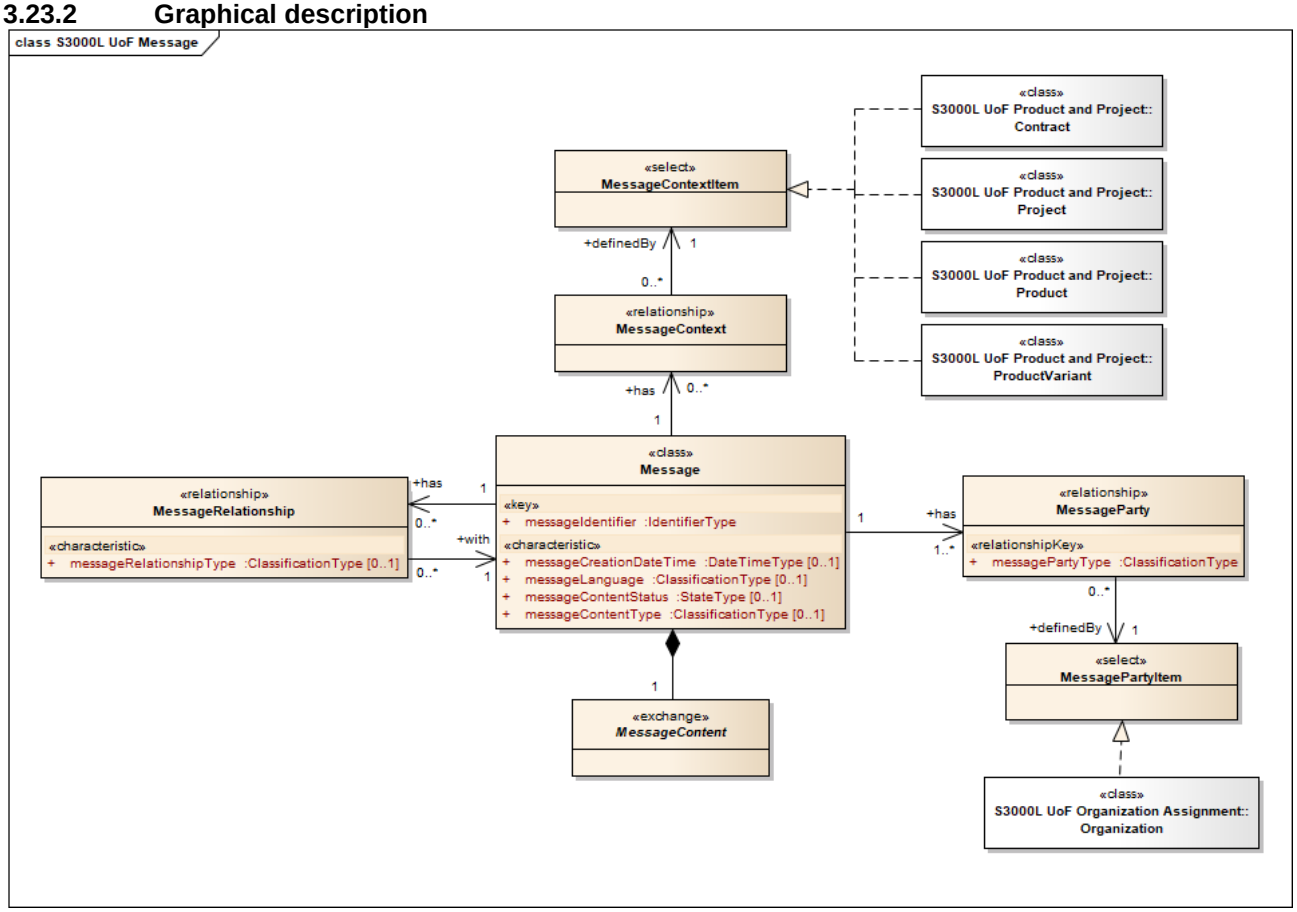
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.23 S3000L UoF Message

3.23.1 Description

The Message UoF provides the capability to identify a collection of information to be communicated from one party to another.

3.23.2 Graphical description



ICN-B6865-S3000L0295-001-01

Fig 27 S3000L UoF Message

3.23.3 Class definition

3.23.3.1 Message

`Message` is a `<<class>>` that represents the collection of information brought together by a message sender for the purpose of communicating it to another party.

3.23.3.1.1 Attribute(s)

This class has the following attributes:

- `messageIdentifier`
- `messageCreationDateTime`, zero or one
- `messageLanguage`, zero or one
- `messageContentStatus`, zero or one
- `messageContentType`, zero or one

3.23.3.1.2 Associations

This class has the following associations:

- An aggregate association with one instance of `MessageContent`
- A directed `has` association with one or many instances of `MessageParty`
- A directed `has` association with zero, one or many instances of `MessageContext`
- A directed `has` association with zero, one or many instances of `MessageRelationship`

3.23.3.1.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem. Refer to Para 3.33

3.23.3.2 MessageContent

MessageContent is a <<exchange>> definition that represents the collection of information that is the subject of the Message.

3.23.3.3 MessageContext

MessageContext is a <<relationship>> between a Message and the context for which it is being provided.

Example(s)

- Contract
- Product
- Project

3.23.3.3.1 Associations

This class has the following associations:

- A directed definedBy association with one instance of MessageContextItem

3.23.3.3.2 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.23.3.4 MessageContextItem

MessageContextItem is a <<select>> interface that identifies items which can be selected as the context for a Message.

3.23.3.4.1 Class members

This <<select>> interface includes the following class members:

- Contract. Refer to Para 3.27
- Product. Refer to Para 3.27
- ProductVariant. Refer to Para 3.27
- Project. Refer to Para 3.27

3.23.3.5 MessageParty

MessageParty is a <<relationship>> between a Message and a stakeholder for the Message.

3.23.3.5.1 Attribute(s)
This class has the following attributes:

- messagePartyType

3.23.3.5.2 Associations
This class has the following associations:

- A directed definedBy association with one instance of MessagePartyItem

3.23.3.5.3 Implementations
This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.23.3.6 MessagePartyItem
MessagePartyItem is a <<select>> interface that identifies items which can be selected as the party for a Message.

3.23.3.6.1 Class members
This <<select>> interface includes the following class members:

- Organization. Refer to Para 3.24

3.23.3.7 MessageRelationship
MessageRelationship is a <<relationship>> where one Message relates to another Message.

Example(s)
- One Message is a reply to another Message
- One Message is an update to another Message

3.23.3.7.1 Attribute(s)
This class has the following attributes:

- messageRelationshipType, zero or one

3.23.3.7.2 Associations
This class has the following associations:

- A directed with association with one instance of Message

3.23.3.7.3 Implementations
This class implements the following <<extend>> interfaces:

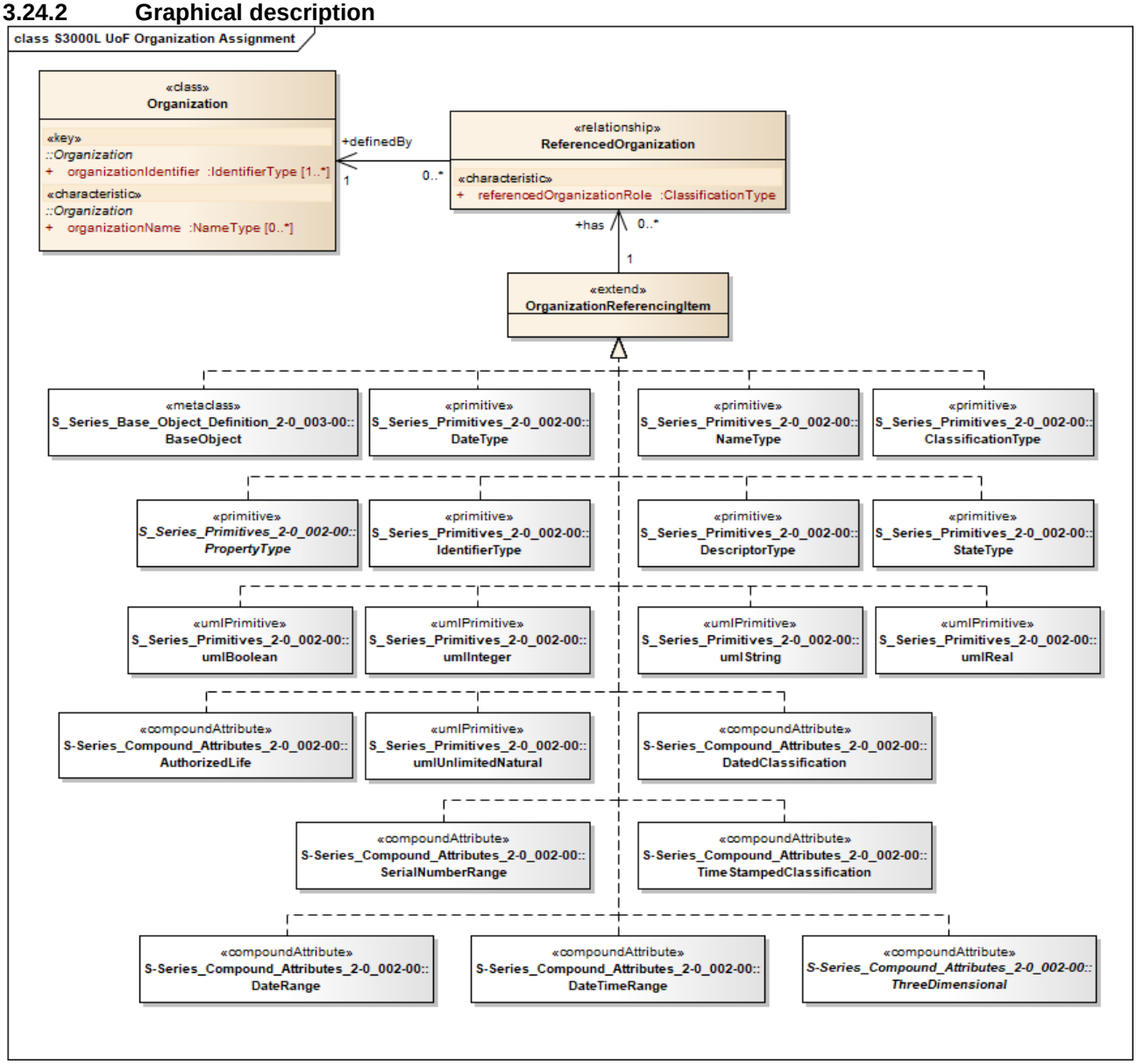
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.24 S3000L UoF Organization Assignment

3.24.1 Description

The Organization Assignment UoF provides the capability to identify and reference organizations.

3.24.2 Graphical description



ICN-B6865-S3000L0232-004-01

Fig 28 S3000L UoF Organization Assignment

3.24.3 Class definition

3.24.3.1 Organization

`Organization` is a `<<class>>` that represents an administrative structure with a particular purpose belonging to a legal entity.

Example(s)

- Government department
- International agency
- Company

- Department

3.24.3.1.1 Attribute(s)

This class has the following attributes:

- organizationIdentifier, one or many
- organizationName, zero, one or many

3.24.3.1.2 Implementations

This class implements the following <<extend>> interfaces:

- DigitalFileReferencingItem. Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.24.3.2 OrganizationReferencingItem

OrganizationReferencingItem is an <<extend>> interface that provides its associated data model to those classes that implement it.

3.24.3.2.1 Class members

Classes that implement the OrganizationReferencingItem <<extend>> interface are:

- AllocatedTaskLocation. Refer to Para 3.39
- AllowedProductConfigurationByConfigurationIdentifier. Refer to Para 3.28
- AnalysisActivity. Refer to Para 3.21
- AnalysisActivityRevision. Refer to Para 3.21
- ApplicabilityStatement. Refer to Para 3.2
- AuthorityToOperate. Refer to Para 3.28
- AuthorizedLife. Refer to SX002D
- Breakdown. Refer to Para 3.3
- BreakdownElement. Refer to Para 3.3
- BreakdownElementRevision. Refer to Para 3.3
- BreakdownElementUsageInBreakdown. Refer to Para 3.3
- BreakdownRevision. Refer to Para 3.3
- ChangeAuthorization. Refer to Para 3.4
- ChangeRequest. Refer to Para 3.9
- CircuitBreaker. Refer to Para 3.5
- CircuitBreakerSetting. Refer to Para 3.36
- CircuitBreakerSettings. Refer to Para 3.36
- ClassificationType. Refer to SX002D
- CompetencyDefinition. Refer to Para 3.6
- ConditionInstance. Refer to Para 3.2
- ConditionType. Refer to Para 3.2
- ConditionTypeAssertMember. Refer to Para 3.2
- Contract. Refer to Para 3.27
- Country. Refer to Para 3.19

- DamageDefinition. Refer to Para 3.7
- DatedClassification. Refer to SX002D
- DateRange. Refer to SX002D
- DateTimeRange. Refer to SX002D
- DateType. Refer to SX002D
- DecisionTreeTemplate. Refer to Para 3.8
- DecisionTreeTemplateActionDefinition. Refer to Para 3.8
- DecisionTreeTemplateQuestionDefinition. Refer to Para 3.8
- DecisionTreeTemplateRevision. Refer to Para 3.8
- DescriptorType. Refer to SX002D
- DetectionMeansAlarm. Refer to Para 3.16
- DigitalFile. Refer to Para 3.10
- Document. Refer to Para 3.11
- DocumentIssue. Refer to Para 3.11
- EnvironmentDefinition. Refer to Para 3.12
- EnvironmentDefinitionCharacteristic. Refer to Para 3.12
- EnvironmentDefinitionRevision. Refer to Para 3.12
- EvaluationCriteria. Refer to Para 3.2
- Facility. Refer to Para 3.13
- FailureMode. Refer to Para 3.14
- FailureModeAnalysis. Refer to Para 3.14
- FailureModeAnalysisRevision. Refer to Para 3.14
- FailureModeCause. Refer to Para 3.14
- FailureModeEffect. Refer to Para 3.14
- FailureModeSymptomsSignature. Refer to Para 3.16
- GeographicalArea. Refer to Para 3.19
- GlobalPosition. Refer to Para 3.19
- IdentifierType. Refer to SX002D
- InServiceOptimizationAnalysis. Refer to Para 3.18
- InServiceOptimizationAnalysisRevision. Refer to Para 3.18
- InServiceOptimizationAnalysisStep. Refer to Para 3.18
- LSAFailureModeGroup. Refer to Para 3.22
- MaintenanceLevel. Refer to Para 3.29
- MeasurementPointDefinition. Refer to Para 3.16
- Message. Refer to Para 3.23
- NameType. Refer to SX002D
- OperatingLocationType. Refer to Para 3.29
- Organization. Refer to Para 3.24
- PartAsDesigned. Refer to Para 3.25
- PartAsDesignedPartsList. Refer to Para 3.25
- PartAsDesignedPartsListEntry. Refer to Para 3.25
- PartAsDesignedPartsListRevision. Refer to Para 3.25
- PerformanceParameter. Refer to Para 3.26
- PerformanceParameterRevision. Refer to Para 3.26
- Product. Refer to Para 3.27
- ProductUsagePhase. Refer to Para 3.30
- ProductVariant. Refer to Para 3.27
- Project. Refer to Para 3.27

- PropertyType. Refer to SX002D
- RepeatDefinition. Refer to Para 3.40
- ResourceSpecification. Refer to Para 3.32
- ResourceSpecificationRevision. Refer to Para 3.32
- SecurityClass. Refer to Para 3.33
- SerialNumberRange. Refer to SX002D
- Skill. Refer to Para 3.6
- SkillLevel. Refer to Para 3.6
- SpecialEventDefinition. Refer to Para 3.35
- StateType. Refer to SX002D
- StreetAddress. Refer to Para 3.19
- SubstanceDefinition. Refer to Para 3.25
- Subtask. Refer to Para 3.36
- Task. Refer to Para 3.36
- TaskRequirement. Refer to Para 3.37
- TaskRequirementRevision. Refer to Para 3.37
- TaskResource. Refer to Para 3.38
- TaskRevision. Refer to Para 3.36
- ThreeDimensional. Refer to SX002D
- ThresholdDefinition. Refer to Para 3.40
- TimeLimit. Refer to Para 3.40
- TimeStampedClassification. Refer to SX002D
- Trade. Refer to Para 3.6
- umlBoolean. Refer to SX002D
- umlInteger. Refer to SX002D
- umlReal. Refer to SX002D
- umlString. Refer to SX002D
- umlUnlimitedNatural. Refer to SX002D
- WarningCautionNote. Refer to Para 3.36

3.24.3.2.2 Associations

The OrganizationReferencingItem <<extend>> interface has the following associations:

- A directed definedBy association with zero, one or many instances of ReferencedOrganization

3.24.3.3 ReferencedOrganization

ReferencedOrganization is a <<relationship>> where one OrganizationReferencingItem relates to an Organization.

3.24.3.3.1 Attribute(s)

This class has the following attributes:

- referencedOrganizationRole

3.24.3.3.2 Associations

This class has the following associations:

- A directed definedBy association with zero, one or many instances of Organization

3.24.3.3.3 Implementations

This class implements the following <<extend>> interfaces:

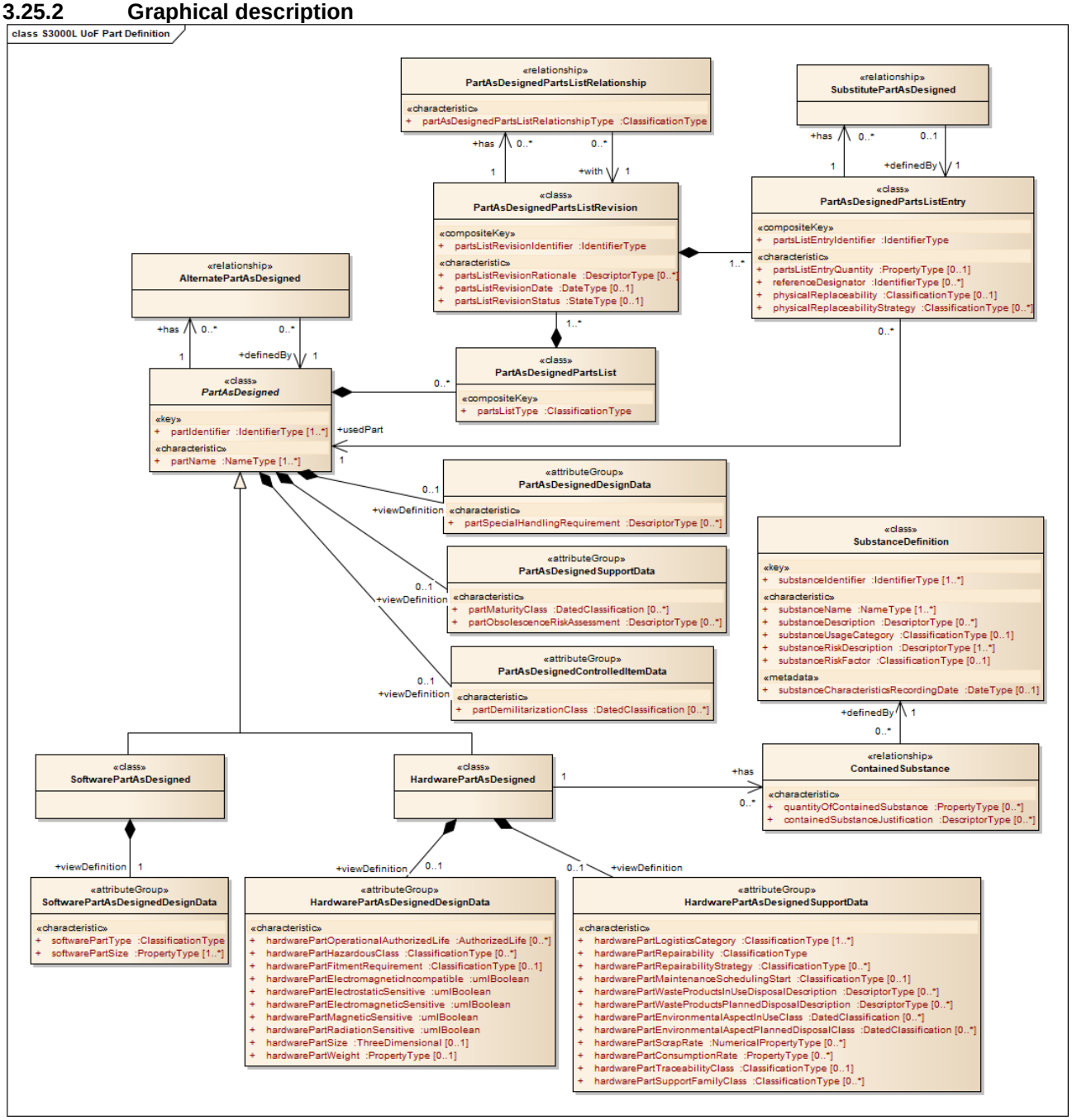
- ApplicabilityStatementItem. Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.25 S3000L UoF Part Definition

3.25.1 Description

The Part Definition UoF provides the capability of defining hardware and software parts, their characteristics, and associated parts lists.

3.25.2 Graphical description



ICN-B6865-S3000L0218-004-01

Fig 29 S3000L UoF Part Definition

3.25.3 Class definition

3.25.3.1 AlternatePartAsDesigned

`AlternatePartAsDesigned` is a `<<relationship>>` that defines an alternate `PartAsDesigned` which can replace the base `PartAsDesigned` in all its usages ie, it is context independent, and is form, fit and function equivalent.

Note

A part can have one or more alternate parts. The alternate part is interchangeable with the base part in any/all uses.

3.25.3.1.1 Associations
This class has the following associations:

- A directed definedBy association with one instance of PartAsDesigned

3.25.3.1.2 Implementations
This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- ChangeControlledItem. Refer to Para 3.4
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.25.3.2 ContainedSubstance
ContainedSubstance is a <<relationship>> that associates a HardwarePartAsDesigned with a contained SubstanceDefinition.

3.25.3.2.1 Attribute(s)
This class has the following attributes:

- quantityOfContainedSubstance, zero, one or many
- containedSubstanceJustification, zero, one or many

3.25.3.2.2 Associations
This class has the following associations:

- A directed definedBy association with one instance of SubstanceDefinition

3.25.3.2.3 Implementations
This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.25.3.3 HardwarePartAsDesigned
HardwarePartAsDesigned is a PartAsDesigned that is to be realized as physical items, including non-countable material.

Note
Examples of non-countable materials are: oil, sealant, paint.

3.25.3.3.1 Attribute(s)
This class has the following attributes:

- partIdentifier (inherited from PartAsDesigned), one or many
- partName (inherited from PartAsDesigned), one or many

3.25.3.3.2 Associations

This class has the following associations:

- An aggregate viewDefinition association with zero or one instance of HardwarePartAsDesignedDesignData
- An aggregate viewDefinition association with zero or one instance of HardwarePartAsDesignedSupportData
- An aggregate viewDefinition association with zero or one instance of PartAsDesignedControlledItemData (inherited from PartAsDesigned)
- An aggregate viewDefinition association with zero or one instance of PartAsDesignedDesignData (inherited from PartAsDesigned)
- An aggregate association with zero, one or many instances of PartAsDesignedPartsList (inherited from PartAsDesigned)
- An aggregate viewDefinition association with zero or one instance of PartAsDesignedSupportData (inherited from PartAsDesigned)
- A directed has association with zero, one or many instances of AlternatePartAsDesigned (inherited from PartAsDesigned)
- A directed has association with zero, one or many instances of ContainedSubstance

3.25.3.3.3 Implementations

This class implements the following <<extend>> interfaces:

- AnalysisCandidateItem (inherited from PartAsDesigned). Refer to Para 3.21
- DecisionTreeAnalysisItem (inherited from PartAsDesigned). Refer to Para 3.8
- DetectionMeansAlarmItem (inherited from PartAsDesigned). Refer to Para 3.16
- DigitalFileReferencingItem (inherited from PartAsDesigned). Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- FailureModeAnalysisItem (inherited from PartAsDesigned). Refer to Para 3.14
- InServiceOptimizationAnalysisItem (inherited from PartAsDesigned). Refer to Para 3.18
- MeasurementPointDefinitionItem (inherited from PartAsDesigned). Refer to Para 3.16
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- PerformanceParameterItem (inherited from PartAsDesigned). Refer to Para 3.26
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem (inherited from PartAsDesigned). Refer to Para 3.33
- TaskAnalysisItem (inherited from PartAsDesigned). Refer to Para 3.39
- TaskRequirementAnalysisItem (inherited from PartAsDesigned). Refer to Para 3.37

3.25.3.4 HardwarePartAsDesignedDesignData

HardwarePartAsDesignedDesignData is an <<attributeGroup>> that collects HardwarePartAsDesigned characteristics identified during design activities.

3.25.3.4.1 Attribute(s)

This <<attributeGroup>> has the following attributes:

- hardwarePartOperationalAuthorizedLife, zero, one or many
- hardwarePartHazardousClass, zero, one or many
- hardwarePartFitmentRequirement, zero or one
- hardwarePartElectromagneticIncompatible
- hardwarePartElectrostaticSensitive
- hardwarePartElectromagneticSensitive
- hardwarePartMagneticSensitive
- hardwarePartRadiationSensitive
- hardwarePartSize, zero or one
- hardwarePartWeight, zero or one

3.25.3.5 HardwarePartAsDesignedSupportData

HardwarePartAsDesignedSupportData is an <<attributeGroup>> that collects HardwarePartAsDesigned characteristics identified during supportability analysis activities.

3.25.3.5.1 Attribute(s)

This <<attributeGroup>> has the following attributes:

- hardwarePartLogisticsCategory, one or many
- hardwarePartRepairability
- hardwarePartRepairabilityStrategy, zero, one or many
- hardwarePartMaintenanceSchedulingStart, zero or one
- hardwarePartWasteProductsInUseDisposalDescription, zero, one or many
- hardwarePartWasteProductsPlannedDisposalDescription, zero, one or many
- hardwarePartEnvironmentalAspectInUseClass, zero, one or many
- hardwarePartEnvironmentalAspectPlannedDisposalClass, zero, one or many
- hardwarePartScrapRate, zero, one or many
- hardwarePartConsumptionRate, zero, one or many
- hardwarePartTraceabilityClass, zero or one
- hardwarePartSupportFamilyClass, zero, one or many

3.25.3.6 PartAsDesigned

PartAsDesigned is a <<class>> that represents the definitional information for an artifact fulfilling a set of requirements, which can be produced or realized.

3.25.3.6.1 Attribute(s)

This class has the following attributes:

- partIdentifier, one or many
- partName, one or many

3.25.3.6.2 Associations

This class has the following associations:

- An aggregate viewDefinition association with zero or one instance of PartAsDesignedControlledItemData
- An aggregate viewDefinition association with zero or one instance of PartAsDesignedDesignData
- An aggregate association with zero, one or many instances of PartAsDesignedPartsList
- An aggregate viewDefinition association with zero or one instance of PartAsDesignedSupportData
- A directed has association with zero, one or many instances of AlternatePartAsDesigned

3.25.3.6.3 Implementations

This class implements the following <<extend>> interfaces:

- AnalysisCandidateItem. Refer to Para 3.21
- DecisionTreeAnalysisItem. Refer to Para 3.8
- DetectionMeansAlarmItem. Refer to Para 3.16
- DigitalFileReferencingItem. Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- FailureModeAnalysisItem. Refer to Para 3.14
- InServiceOptimizationAnalysisItem. Refer to Para 3.18
- MeasurementPointDefinitionItem. Refer to Para 3.16
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- PerformanceParameterItem. Refer to Para 3.26
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem. Refer to Para 3.33
- TaskAnalysisItem. Refer to Para 3.39
- TaskRequirementAnalysisItem. Refer to Para 3.37

3.25.3.7 PartAsDesignedControlledItemData

PartAsDesignedControlledItemData is an <<attributeGroup>> that collects PartAsDesigned characteristics identified during different controlled item analysis activities.

3.25.3.7.1 Attribute(s)

This class has the following attributes:

- partDemilitarizationClass, zero, one or many

3.25.3.8 PartAsDesignedDesignData

PartAsDesignedDesignData is an <<attributeGroup>> that collects PartAsDesigned characteristics identified during design activities.

3.25.3.8.1 Attribute(s)

This class has the following attributes:

- partSpecialHandlingRequirement, zero, one or many

3.25.3.9 PartAsDesignedPartsList

PartAsDesignedPartsList is a <<class>> that represents the definitional information for the collection of PartAsDesignedPartsListEntry included in the assembly of the parent PartAsDesigned.

Note

PartAsDesignedPartsList is typically referred to as a Bill of Material (BOM).

3.25.3.9.1 Attribute(s)

This class has the following attributes:

- partsListType

3.25.3.9.2 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of PartAsDesignedPartsListRevision

3.25.3.9.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.25.3.10 PartAsDesignedPartsListEntry

PartAsDesignedPartsListEntry is a <<class>> that represents the inclusion of a PartAsDesigned in a PartAsDesignedPartsListRevision.

3.25.3.10.1 Attribute(s)

This class has the following attributes:

- partsListEntryIdentifier
- partsListEntryQuantity, zero or one
- referenceDesignator, zero, one or many
- physicalReplaceability, zero or one
- physicalReplaceabilityStrategy, zero, one or many

3.25.3.10.2 Associations

This class has the following associations:

- A directed has association with zero, one or many instances of SubstitutePartAsDesigned
- A directed usedPart association with one instance of PartAsDesigned

3.25.3.10.3 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- ChangeControlledItem. Refer to Para 3.4
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- EffectiveOnProductConfigurationItem. Refer to Para 3.28

- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- UsableOnItem. Refer to Para 3.28

3.25.3.11 PartAsDesignedPartsListRelationship

PartAsDesignedPartsListRelationship is a <<relationship>> where one PartAsDesignedPartsList relates to another PartAsDesignedPartsList.

3.25.3.11.1 Attribute(s)

This class has the following attributes:

- partAsDesignedPartsListRelationshipType

3.25.3.11.2 Associations

This class has the following associations:

- A directed with association with one instance of PartAsDesignedPartsListRevision

3.25.3.12 PartAsDesignedPartsListRevision

PartAsDesignedPartsListRevision is a <<class>> representing an iteration applied to a PartAsDesignedPartsList.

3.25.3.12.1 Attribute(s)

This class has the following attributes:

- partsListRevisionIdentifier
- partsListRevisionRationale, zero, one or many
- partsListRevisionDate, zero or one
- partsListRevisionStatus, zero or one

3.25.3.12.2 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of PartAsDesignedPartsListEntry
- A directed has association with zero, one or many instances of PartAsDesignedPartsListRelationship

3.25.3.12.3 Implementations

This class implements the following <<extend>> interfaces:

- ChangeControlledItem. Refer to Para 3.4
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- EffectiveOnProductConfigurationItem. Refer to Para 3.28
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- UsableOnItem. Refer to Para 3.28

3.25.3.13 PartAsDesignedSupportData

PartAsDesignedSupportData is an <<attributeGroup>> that collects PartAsDesigned characteristics identified during supportability analysis activities.

3.25.3.13.1 Attribute(s)

This <<attributeGroup>> has the following attributes:

- partMaturityClass, zero, one or many
- partObsolescenceRiskAssessment, zero, one or many

3.25.3.14 SoftwarePartAsDesigned

SoftwarePartAsDesigned is a PartAsDesigned that is produced as an executable software or as a data file.

Note

Non-executable software includes eg, maps.

3.25.3.14.1 Attribute(s)

This class has the following attributes:

- partIdentifier (inherited from PartAsDesigned), one or many
- partName (inherited from PartAsDesigned), one or many

3.25.3.14.2 Associations

This class has the following associations:

- An aggregate viewDefinition association with zero or one instance of PartAsDesignedControlledItemData (inherited from PartAsDesigned)
- An aggregate viewDefinition association with zero or one instance of PartAsDesignedDesignData (inherited from PartAsDesigned)
- An aggregate association with zero, one or many instances of PartAsDesignedPartsList (inherited from PartAsDesigned)
- An aggregate viewDefinition association with zero or one instance of PartAsDesignedSupportData (inherited from PartAsDesigned)
- A directed has association with zero, one or many instances of AlternatePartAsDesigned (inherited from PartAsDesigned)

3.25.3.14.3 Implementations

This class implements the following <<extend>> interfaces:

- AnalysisCandidateItem (inherited from PartAsDesigned). Refer to Para 3.21
- DecisionTreeAnalysisItem (inherited from PartAsDesigned). Refer to Para 3.8
- DetectionMeansAlarmItem (inherited from PartAsDesigned). Refer to Para 3.16
- DigitalFileReferencingItem (inherited from PartAsDesigned). Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- FailureModeAnalysisItem (inherited from PartAsDesigned). Refer to Para 3.14
- InServiceOptimizationAnalysisItem (inherited from PartAsDesigned). Refer to Para 3.18
- MeasurementPointDefinitionItem (inherited from PartAsDesigned). Refer to Para 3.16

- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- PerformanceParameterItem (inherited from PartAsDesigned). Refer to Para 3.26
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem (inherited from PartAsDesigned). Refer to Para 3.33
- TaskAnalysisItem (inherited from PartAsDesigned). Refer to Para 3.39
- TaskRequirementAnalysisItem (inherited from PartAsDesigned). Refer to Para 3.37

3.25.3.15 SoftwarePartAsDesignedDesignData

SoftwarePartAsDesignedDesignData is an <<attributeGroup>> that collects SoftwarePartAsDesigned characteristics identified during design activities.

3.25.3.15.1 Attribute(s)

This <<attributeGroup>> has the following attributes:

- softwarePartType
- softwarePartSize, one or many

3.25.3.16 SubstanceDefinition

SubstanceDefinition is a <<class>> that identifies high concern physical matter.

Example(s)

- CAS (Chemical Abstract Substance) Registry
- REACH (Registration, Evaluation, Authorisation and Restriction of Chemicals)

3.25.3.16.1 Attribute(s)

This class has the following attributes:

- substanceIdentifier, one or many
- substanceName, one or many
- substanceDescription, zero, one or many
- substanceUsageCategory, zero or one
- substanceRiskDescription, one or many
- substanceRiskFactor, zero or one
- substanceCharacteristicsRecordingDate, zero or one

3.25.3.16.2 Implementations

This class implements the following <<extend>> interfaces:

- DigitalFileReferencingItem. Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.25.3.17 SubstitutePartAsDesigned

SubstitutePartAsDesigned is a <<relationship>> that defines a substitute PartAsDesignedPartsListEntry which can replace the base PartAsDesignedPartsListEntry in the context of the parent PartAsDesignedPartsList.

3.25.3.17.1 Associations

This class has the following associations:

- A directed definedBy association with one instance of PartAsDesignedPartsListEntry

3.25.3.17.2 Implementations

This class implements the following <<extend>> interfaces:

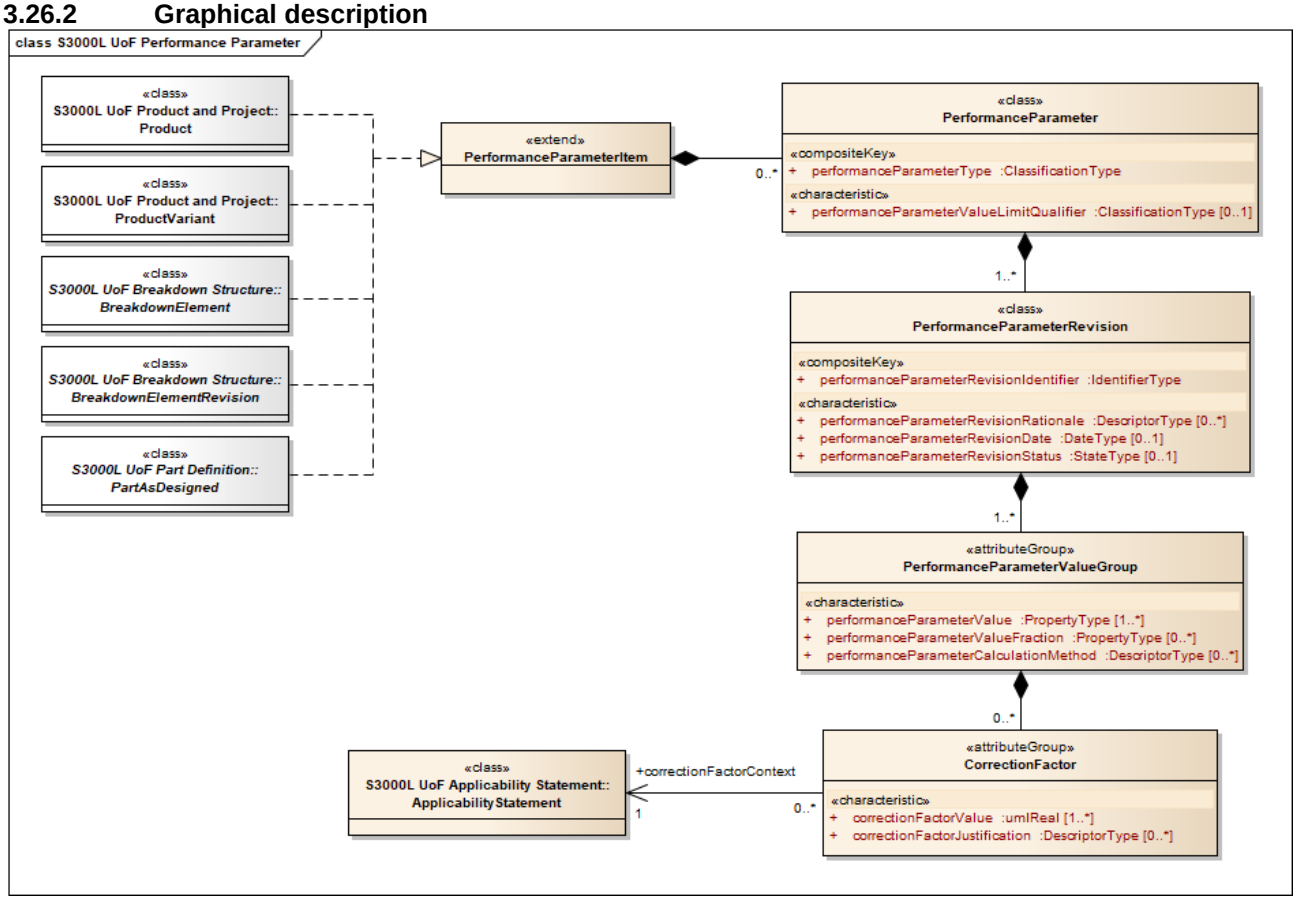
- ChangeControlledItem. Refer to Para 3.4
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.26 S3000L UoF Performance Parameter

3.26.1 Description

The Performance Parameter UoF supports the definition of metrics that if changed will have a major impact on the system performance, schedule, cost and/or risk.

3.26.2 Graphical description



ICN-B6865-S3000L0275-001-01

Fig 30 S3000L UoF Performance Parameter

3.26.3 Class definition

3.26.3.1 CorrectionFactor

`CorrectionFactor` is an `<<attributeGroup>>` that defines a mathematical adjustment of a defined value to account for a specified deviation in a given context.

3.26.3.1.1 Attribute(s)

This `<<attributeGroup>>` has the following attributes:

- `correctionFactorValue`, one or many
- `correctionFactorJustification`, zero, one or many

3.26.3.1.2 Associations

This `<<attributeGroup>>` has the following associations:

- A directed `correctionFactorContext` association with one instance of `ApplicabilityStatement`

3.26.3.2 PerformanceParameter

`PerformanceParameter` is a `<<class>>` that represents a metric that if changed, or not fulfilled, can have a major impact on the performance, schedule, cost and/or risk for the `PerformanceParameterItem`.

3.26.3.2.1 Attribute(s)

This class has the following attributes:

- performanceParameterType
- performanceParameterValueLimitQualifier, zero or one

3.26.3.2.2 Associations

This class has the following associations:

- An aggregate association with one or many instances of PerformanceParameterRevision

3.26.3.2.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem. Refer to Para 3.33

3.26.3.3 PerformanceParameterItem

PerformanceParameterItem is an <<extend>> interface that provides its associated data model to those classes that can have an associated PerformanceParameter.

3.26.3.3.1 Class members

Classes that implement the PerformanceParameterItem <<extend>> interface are:

- BreakdownElement. Refer to Para 3.3
- BreakdownElementRevision. Refer to Para 3.3
- PartAsDesigned. Refer to Para 3.25
- Product. Refer to Para 3.27
- ProductVariant. Refer to Para 3.27

3.26.3.3.2 Associations

The PerformanceParameterItem <<extend>> interface has the following associations:

- An aggregate association with zero, one or many instances of PerformanceParameter

3.26.3.4 PerformanceParameterRevision

PerformanceParameterRevision is a <<class>> that represents an iteration applied to a PerformanceParameter.

3.26.3.4.1 Attribute(s)

This class has the following attributes:

- performanceParameterRevisionIdentifier
- performanceParameterRevisionRationale, zero, one or many
- performanceParameterRevisionDate, zero or one
- performanceParameterRevisionStatus, zero or one

3.26.3.4.2 Associations

This class has the following associations:

- An aggregate association with one or many instances of PerformanceParameterValueGroup

3.26.3.4.3 Implementations

This class implements the following <<extend>> interfaces:

- ChangeControlledItem. Refer to Para 3.4
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.26.3.5 PerformanceParameterValueGroup

PerformanceParameterValueGroup is an <<attributeGroup>> that organizes PerformanceParameter values for a defined purpose.

3.26.3.5.1 Attribute(s)

This <<attributeGroup>> has the following attributes:

- performanceParameterValue, one or many
- performanceParameterValueFraction, zero, one or many
- performanceParameterCalculationMethod, zero, one or many

3.26.3.5.2 Associations

This <<attributeGroup>> has the following associations:

- An aggregate association with zero, one or many instances of CorrectionFactor

3.26.3.5.3 Implementations

This class implements the following <<extend>> interfaces:

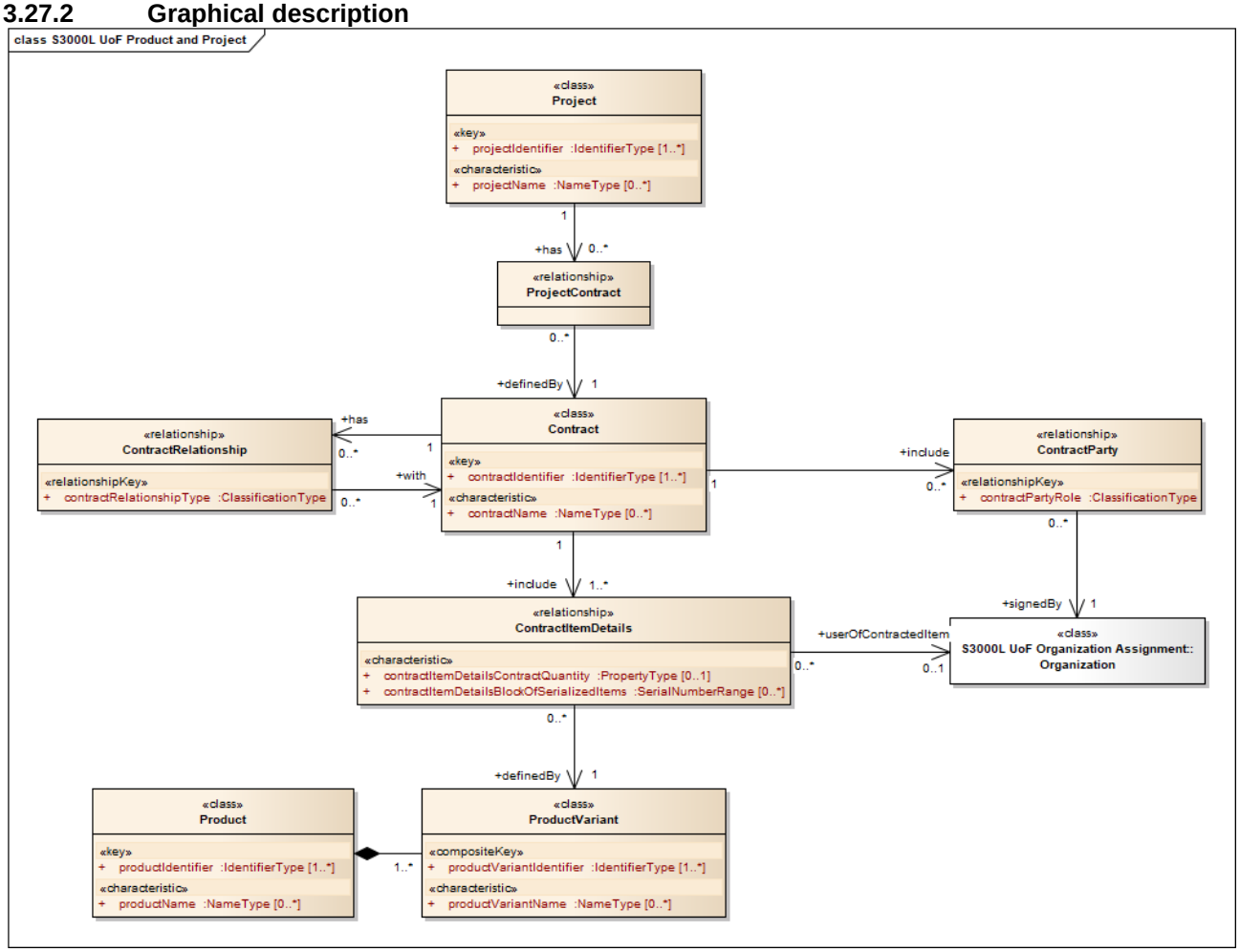
- ApplicabilityStatementItem. Refer to Para 3.2

3.27 S3000L UoF Product and Project

3.27.1 Description

The Product and Project UoF defines the Product(s) which are in focus for the Integrated Product Support (IPS) program.

3.27.2 Graphical description



ICN-B6865-S3000L0215-004-01

Fig 31 S3000L UoF Product and Project

3.27.3 Class definition

3.27.3.1 Contract

`Contract` is a `<<class>>` that represents a binding agreement between two or more parties.

3.27.3.1.1 Attribute(s)

This class has the following attributes:

- `contractIdentifier`, one or many
- `contractName`, zero, one or many

3.27.3.1.2 Associations

This class has the following associations:

- A directed `include` association with one or many instances of `ContractItemDetails`
- A directed `include` association with zero, one or many instances of `ContractParty`
- A directed `has` association with zero, one or many instances of `ContractRelationship`

3.27.3.1.3 Implementations

This class implements the following <<extend>> interfaces:

- DigitalFileReferencingItem. Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem. Refer to Para 3.33

3.27.3.1.4 Selects

This class is a member of the following <<select>> interfaces:

- ClassInstanceAssertItem. Refer to Para 3.2
- MessageContextItem. Refer to Para 3.23

3.27.3.2 ContractItemDetails

ContractItemDetails is a <<relationship>> that identifies an item which is the subject of the Contract.

3.27.3.2.1 Attribute(s)

This class has the following attributes:

- contractItemDetailsContractQuantity, zero or one
- contractItemDetailsBlockOfSerializedItems, zero, one or many

3.27.3.2.2 Associations

This class has the following associations:

- A directed definedBy association with one instance of ProductVariant
- A directed userOfContractedItem association with zero or one instance of Organization

3.27.3.3 ContractParty

ContractParty is a <<relationship>> that identifies a Contract stakeholder.

3.27.3.3.1 Attribute(s)

This class has the following attributes:

- contractPartyRole

3.27.3.3.2 Associations

This class has the following associations:

- A directed signedBy association with one instance of Organization

3.27.3.3.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.27.3.4 ContractRelationship

ContractRelationship is a <<relationship>> where one Contract relates to another Contract.

3.27.3.4.1 Attribute(s)

This class has the following attributes:

- contractRelationshipType

3.27.3.4.2 Associations

This class has the following associations:

- A directed with association with one instance of Contract

3.27.3.4.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.27.3.5 Product

Product is a <<class>> that represents a family of items which share the same underlying design purpose.

Example(s)

- Aegis Class Destroyer
- Airbus A340
- Ford Fusion
- iPhone 12
- Pegasus engine
- Stryker

3.27.3.5.1 Attribute(s)

This class has the following attributes:

- productIdentifier, one or many
- productName, zero, one or many

3.27.3.5.2 Associations

This class has the following associations:

- An aggregate association with one or many instances of ProductVariant

3.27.3.5.3 Implementations

This class implements the following <<extend>> interfaces:

- AnalysisCandidateItem. Refer to Para 3.21
- BreakdownItem. Refer to Para 3.3
- DecisionTreeAnalysisItem. Refer to Para 3.8
- DetectionMeansAlarmItem. Refer to Para 3.16
- DigitalFileReferencingItem. Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- InServiceOptimizationAnalysisItem. Refer to Para 3.18

- MeasurementPointDefinitionItem. Refer to Para 3.16
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- PerformanceParameterItem. Refer to Para 3.26
- ProductUsagePhaseItem. Refer to Para 3.30
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem. Refer to Para 3.33

3.27.3.6 ProductVariant

ProductVariant is a <<class>> that defines a member of a Product family which is configured for a specific purpose and is made available to the market.

Note

A product variant is often known as a model.

Example(s)

- Boeing 787-800 vs 787-900
- Ford Fusion S vs. SE vs. SEL

3.27.3.6.1 Attribute(s)

This class has the following attributes:

- productVariantIdentifier, one or many
- productVariantName, zero, one or many

3.27.3.6.2 Associations

This class has the following associations:

- A directed operatedAt association with zero, one or many instances of ContractedProductVariantAtOperatingLocation. Refer to Para 3.29
- A directed operatedAt association with zero, one or many instances of ContractedProductVariantAtOperatingLocationType. Refer to Para 3.29
- A directed include association with zero, one or many instances of NestedProductVariant. Refer to Para 3.28

3.27.3.6.3 Implementations

This class implements the following <<extend>> interfaces:

- AnalysisCandidateItem. Refer to Para 3.21
- BreakdownItem. Refer to Para 3.3
- DecisionTreeAnalysisItem. Refer to Para 3.8
- DetectionMeansAlarmItem. Refer to Para 3.16
- DigitalFileReferencingItem. Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- InServiceOptimizationAnalysisItem. Refer to Para 3.18
- MeasurementPointDefinitionItem. Refer to Para 3.16
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- PerformanceParameterItem. Refer to Para 3.26
- ProductUsagePhaseItem. Refer to Para 3.30

- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem. Refer to Para 3.33

3.27.3.7 Project

Project is a <<class>> that represents the overall set of Integrated Product Support (IPS) activities defined for a Product.

Note

Project is often referred to as an IPS program.

3.27.3.7.1 Attribute(s)

This class has the following attributes:

- projectIdentifier, one or many
- projectName, zero, one or many

3.27.3.7.2 Associations

This class has the following associations:

- A directed has association with zero, one or many instances of ProjectContract

3.27.3.7.3 Implementations

This class implements the following <<extend>> interfaces:

- DecisionTreeAnalysisItem. Refer to Para 3.8
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- InServiceOptimizationAnalysisItem. Refer to Para 3.18
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem. Refer to Para 3.33

3.27.3.8 ProjectContract

ProjectContract is a <<relationship>> that establishes an association between a Project and a Contract.

3.27.3.8.1 Associations

This class has the following associations:

- A directed definedBy association with one instance of Contract

3.27.3.8.2 Implementations

This class implements the following <<extend>> interfaces:

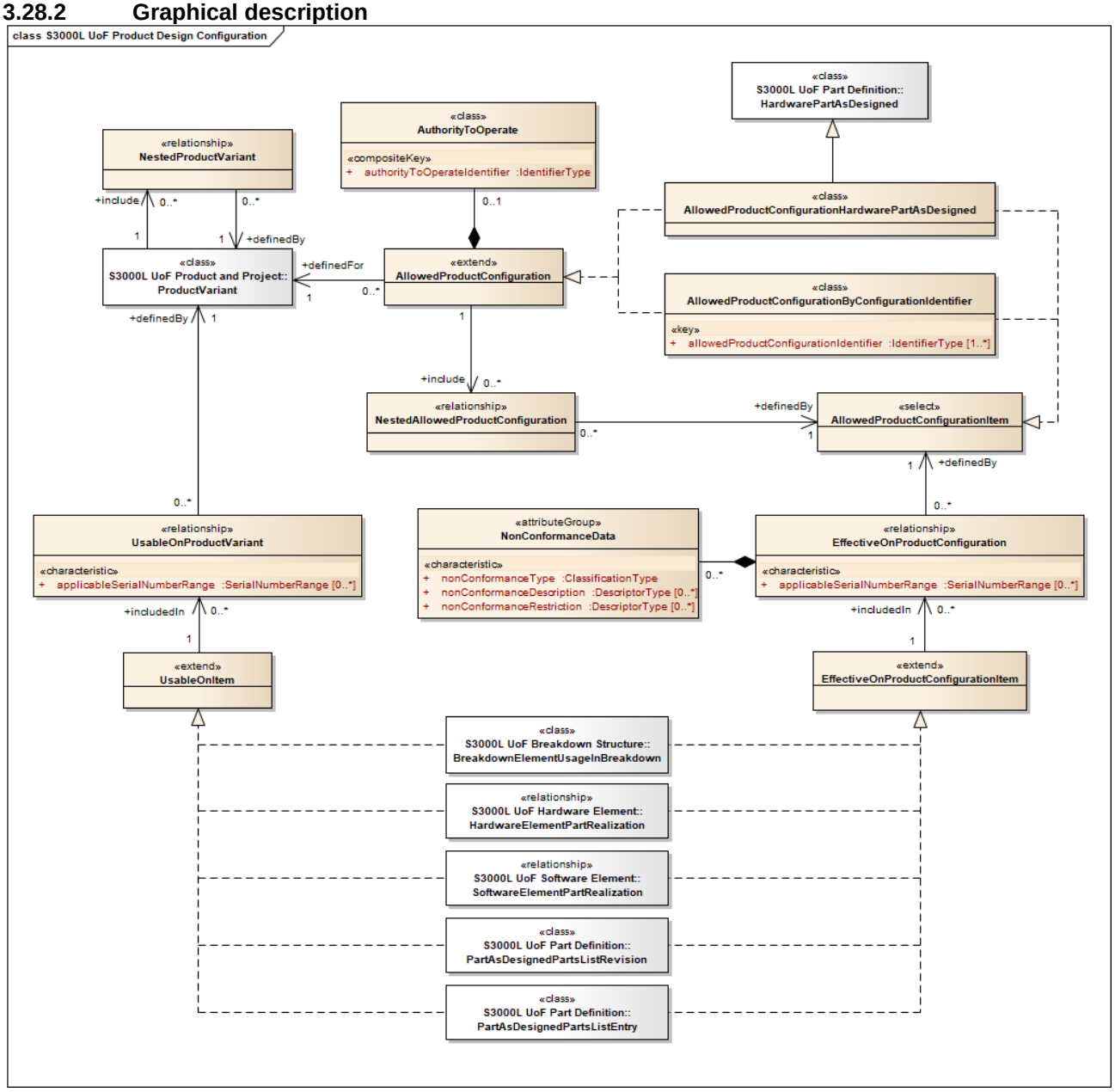
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.28 S3000L UoF Product Design Configuration

3.28.1 Description

The Product Design Configuration UoF defines permitted combinations of breakdown elements, hardware and software, in the context of product variants and allowed product configurations.

3.28.2 Graphical description



ICN-B6865-S3000L0222-004-01

Fig 32 S3000L UoF Product Design Configuration

3.28.3 Class definition

3.28.3.1 AllowedProductConfiguration

`AllowedProductConfiguration` is an `<<extend>>` interface that provides its associated data model to those classes that must define permitted combinations of hardware and software parts which can or must be installed in specific locations (positions).

Note
One and the same serialized product can adhere to different allowed product configurations over time.

Note
An allowed product configuration can also include associated engineering instructions that must be adhered to during assembly and operation and demonstrates that a product complies with applicable regulations.

Example(s)
– Applicable regulations may be a type certificate.

3.28.3.1.1 Class members
Classes that implement the AllowedProductConfiguration <<extend>> interface are:

– AllowedProductConfigurationByConfigurationIdentifier
– AllowedProductConfigurationHardwarePartAsDesigned

3.28.3.1.2 Associations
The AllowedProductConfiguration <<extend>> interface has the following associations:

– A directed include association with zero, one or many instances of NestedAllowedProductConfiguration
– A directed definedFor association with one instance of ProductVariant

3.28.3.2 AllowedProductConfigurationByConfigurationIdentifier
AllowedProductConfigurationByConfigurationIdentifier is a <<class>> that defines an AllowedProductConfiguration by means other than a part number.

3.28.3.2.1 Attribute(s)
This class has the following attributes:

– allowedProductConfigurationIdentifier, one or many

3.28.3.2.2 Implementations
This class implements the following <<extend>> interfaces:

– AllowedProductConfiguration
– DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
– OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
– ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
– RemarkItem (inherited from BaseObject). Refer to Para 3.31
– SecurityClassificationItem. Refer to Para 3.33

3.28.3.3 AllowedProductConfigurationHardwarePartAsDesigned
AllowedProductConfigurationHardwarePartAsDesigned is a HardwarePartAsDesigned (refer to Para 3.25) that is managed as an AllowedProductConfiguration.

3.28.3.3.1 Attribute(s)
This class has the following attributes:

- partIdentifier (inherited from PartAsDesigned), one or many
- partName (inherited from PartAsDesigned), one or many

3.28.3.3.2 Associations

This class has the following associations:

- An aggregate viewDefinition association with zero or one instance of HardwarePartAsDesignedDesignData (inherited from HardwarePartAsDesigned)
- An aggregate viewDefinition association with zero or one instance of HardwarePartAsDesignedSupportData (inherited from HardwarePartAsDesigned)
- An aggregate viewDefinition association with zero or one instance of PartAsDesignedControlledItemData (inherited from PartAsDesigned)
- An aggregate viewDefinition association with zero or one instance of PartAsDesignedDesignData (inherited from PartAsDesigned)
- An aggregate association with zero, one or many instances of PartAsDesignedPartsList (inherited from PartAsDesigned)
- An aggregate viewDefinition association with zero or one instance of PartAsDesignedSupportData (inherited from PartAsDesigned)
- A directed has association with zero, one or many instances of AlternatePartAsDesigned (inherited from PartAsDesigned)
- A directed has association with zero, one or many instances of ContainedSubstance

3.28.3.3.3 Implementations

This class implements the following <<extend>> interfaces:

- AllowedProductConfiguration
- AnalysisCandidateItem (inherited from PartAsDesigned). Refer to Para 3.21
- DecisionTreeAnalysisItem (inherited from PartAsDesigned). Refer to Para 3.8
- DetectionMeansAlarmItem (inherited from PartAsDesigned). Refer to Para 3.16
- DigitalFileReferencingItem (inherited from PartAsDesigned). Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- FailureModeAnalysisItem (inherited from PartAsDesigned). Refer to Para 3.14
- InServiceOptimizationAnalysisItem (inherited from PartAsDesigned). Refer to Para 3.18
- MeasurementPointDefinitionItem (inherited from PartAsDesigned). Refer to Para 3.16
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- PerformanceParameterItem (inherited from PartAsDesigned). Refer to Para 3.26
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

- SecurityClassificationItem (inherited from PartAsDesigned). Refer to Para 3.33
- TaskAnalysisItem (inherited from PartAsDesigned). Refer to Para 3.39
- TaskRequirementAnalysisItem (inherited from PartAsDesigned). Refer to Para 3.37

3.28.3.4 AllowedProductConfigurationItem

AllowedProductConfigurationItem is a <<select>> interface that identifies items which can be selected as an allowed product configuration.

3.28.3.4.1 Class members

This <<select>> interface includes the following class members:

- AllowedProductConfigurationByConfigurationIdentifier
- AllowedProductConfigurationHardwarePartAsDesigned

3.28.3.5 AuthorityToOperate

AuthorityToOperate is a <<class>> that represents a certification allowing a specific configuration of a product to be put into operation.

Note

A design change cannot be put into operation without re-certification.

Note

Type certificate for an aircraft signifies the airworthiness of its design.

3.28.3.5.1 Attribute(s)

This class has the following attributes:

- authorityToOperateIdentifier

3.28.3.5.2 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.28.3.6 EffectiveOnProductConfiguration

EffectiveOnProductConfiguration is a <<relationship>> that identifies that a EffectiveOnProductConfigurationItem, included in the Breakdown (refer to Para 3.3) for the overall Product (refer to Para 3.27), is effective in the associated AllowedProductConfiguration.

3.28.3.6.1 Attribute(s)

This class has the following attributes:

- applicableSerialNumberRange, zero, one or many

3.28.3.6.2 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of NonConformanceData

- A directed definedBy association with one instance of AllowedProductConfigurationItem

3.28.3.6.3 Implementations

This class implements the following <<extend>> interfaces:

- ChangeControlledItem. Refer to Para 3.4
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.28.3.7 EffectiveOnProductConfigurationItem

EffectiveOnProductConfigurationItem is an <<extend>> interface that provides its associated data model to those classes that can be included in one or many instances of AllowedProductConfiguration.

3.28.3.7.1 Class members

Classes that implement the EffectiveOnProductConfigurationItem <<extend>> interface are:

- BreakdownElementUsageInBreakdown. Refer to Para 3.3
- HardwareElementPartRealization. Refer to Para 3.17
- PartAsDesignedPartsListEntry. Refer to Para 3.25
- PartAsDesignedPartsListRevision. Refer to Para 3.25
- SoftwareElementPartRealization. Refer to Para 3.34

3.28.3.7.2 Associations

The EffectiveOnProductConfigurationItem <<extend>> interface has the following associations:

- A directed includedIn association with zero, one or many instances of EffectiveOnProductConfiguration

3.28.3.8 NestedAllowedProductConfiguration

NestedAllowedProductConfiguration is a <<relationship>> that defines that one AllowedProductConfiguration includes a subordinate AllowedProductConfiguration.

3.28.3.8.1 Associations

This class has the following associations:

- A directed definedBy association with one instance of AllowedProductConfigurationItem

3.28.3.8.2 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D

- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.28.3.9 NestedProductVariant

NestedProductVariant is a <<relationship>> that defines that one ProductVariant (refer to Para 3.27) includes a subordinate ProductVariant.

3.28.3.9.1 Associations

This class has the following associations:

- A directed definedBy association with zero, one or many instances of ProductVariant. Refer to Para 3.27

3.28.3.9.2 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.28.3.10 NonConformanceData

NonConformanceData is an <<attributeGroup>> that collects information on how the EffectiveOnProductConfigurationItem does not comply with the requirements of its usage.

3.28.3.10.1 Attribute(s)

This <<attributeGroup>> has the following attributes:

- nonConformanceType
- nonConformanceDescription, zero, one or many
- nonConformanceRestriction, zero, one or many

3.28.3.11 UsableOnItem

UsableOnItem is an <<extend>> interface that provides its associated data model to those classes that can have a limited effectivity with respect to its usage in one or many instances of ProductVariant (refer to Para 3.27).

3.28.3.11.1 Class members

Classes that implement the UsableOnItem <<extend>> interface are:

- BreakdownElementUsageInBreakdown. Refer to Para 3.3
- HardwareElementPartRealization. Refer to Para 3.17
- PartAsDesignedPartsListEntry. Refer to Para 3.25
- PartAsDesignedPartsListRevision. Refer to Para 3.25
- SoftwareElementPartRealization. Refer to Para 3.34

3.28.3.11.2 Associations

The UsableOnItem <<extend>> interface has the following associations:

- A directed includedIn association with zero, one or many instances of UsableOnProductVariant

3.28.3.12 UsableOnProductVariant

UsableOnProductVariant is a <<relationship>> that defines that a UsableOnItem, included in the Breakdown (refer to Para 3.3) for the overall Product (refer to Para 3.27), is effective in the associated ProductVariant (refer to Para 3.27).

Note

UsableOnProductVariant is the equivalent of the Usable On Code in GEIA-0007.

3.28.3.12.1 Attribute(s)

This class has the following attributes:

- applicableSerialNumberRange, zero, one or many

3.28.3.12.2 Associations

This class has the following associations:

- A directed definedBy association with zero, one or many instances of ProductVariant. Refer to Para 3.27

3.28.3.12.3 Implementations

This class implements the following <<extend>> interfaces:

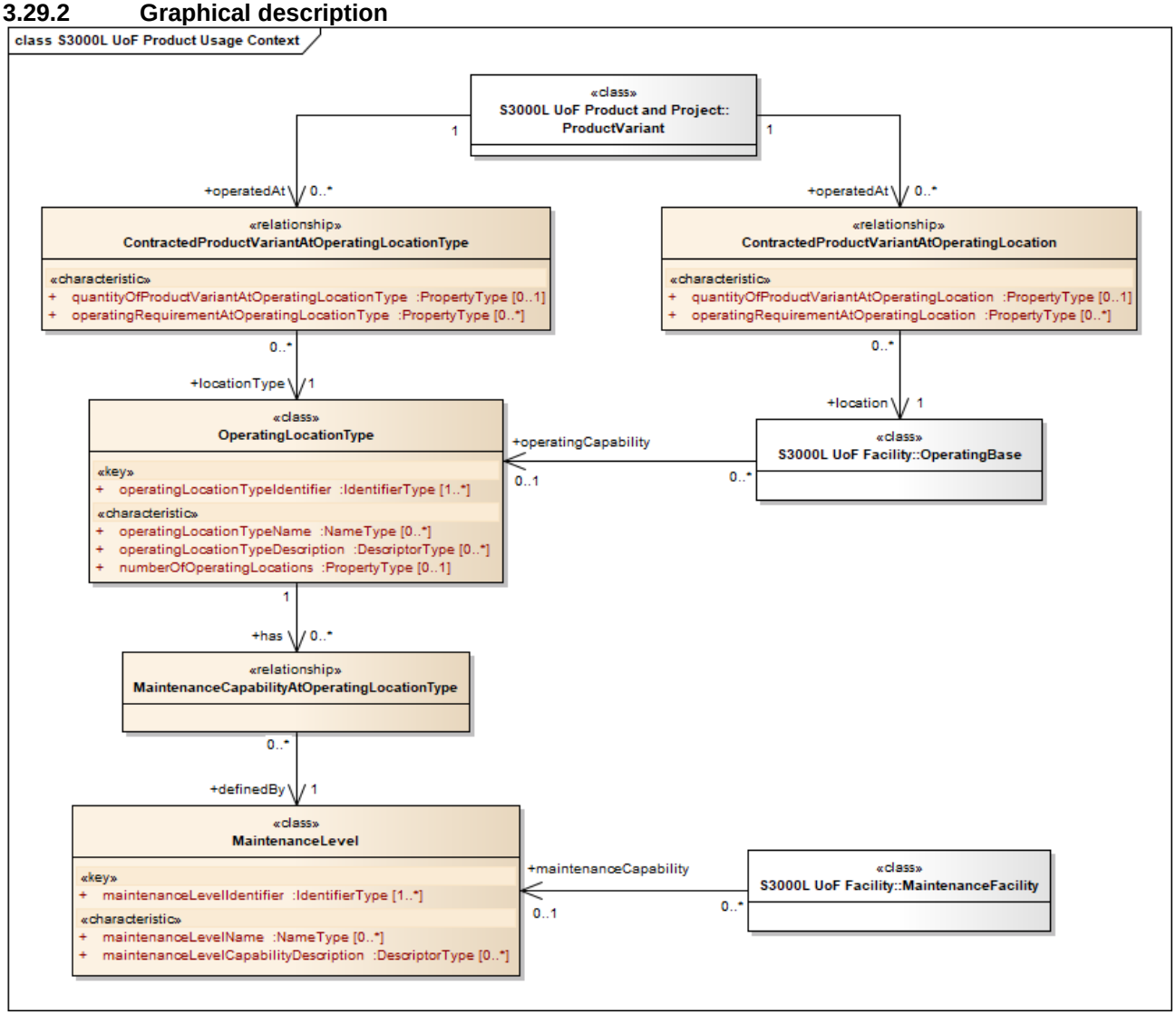
- ChangeControlledItem. Refer to Para 3.4
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.29 S3000L UoF Product Usage Context

3.29.1 Description

The Product Usage Context UoF defines the context in which the defined Product(s) and Product variant(s) are to be operated and maintained.

3.29.2 Graphical description



ICN-B6865-S3000L0216-003-01

Fig 33 S3000L UoF Product Usage Context

3.29.3 Class definition

3.29.3.1 ContractedProductVariantAtOperatingLocation

`ContractedProductVariantAtOperatingLocation` is a `<<relationship>>` that identifies an operating location where a contracted `ProductVariant` (refer to Para 3.27) is defined to be operated.

3.29.3.1.1 Attribute(s)

This class has the following attributes:

- `quantityOfProductVariantAtOperatingLocation`, zero or one
- `operatingRequirementAtOperatingLocation`, zero, one or many

3.29.3.1.2 Associations

This class has the following associations:

- A directed `location` association with one instance of `OperatingBase`. Refer to Para 3.13

3.29.3.1.3 Implementations
This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.29.3.2 ContractedProductVariantAtOperatingLocationType
ContractedProductVariantAtOperatingLocationType is a <<relationship>> that identifies an OperatingLocationType where a contracted ProductVariant (refer to Para 3.27) is defined to be operated.

3.29.3.2.1 Attribute(s)
This class has the following attributes:

- quantityOfProductVariantAtOperatingLocationType, zero or one
- operatingRequirementAtOperatingLocationType, zero, one or many

3.29.3.2.2 Associations
This class has the following associations:

- A directed locationType association with one instance of OperatingLocationType

3.29.3.2.3 Implementations
This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.29.3.3 MaintenanceCapabilityAtOperatingLocationType
MaintenanceCapabilityAtOperatingLocationType is a <<relationship>> that identifies maintenance capabilities which are available at the defined OperatingLocationType.

3.29.3.3.1 Associations
This class has the following associations:

- A directed definedBy association with one instance of MaintenanceLevel

3.29.3.3.2 Implementations
This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.29.3.4 MaintenanceLevel

MaintenanceLevel is a <<class>> that represents the definition of a set of maintenance capabilities which will be made available to support a defined Product (refer to Para 3.27).

Note

MaintenanceLevel might be established either by a single organization or be distributed between a set of organizations.

3.29.3.4.1 Attribute(s)

This class has the following attributes:

- maintenanceLevelIdentifier, one or many
- maintenanceLevelName, zero, one or many
- maintenanceLevelCapabilityDescription, zero, one or many

3.29.3.4.2 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- EnvironmentDefinitionItem. Refer to Para 3.12
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.29.3.5 OperatingLocationType

OperatingLocationType is a <<class>> that represents the definition of the nature of the environment in which a product will be operated.

3.29.3.5.1 Attribute(s)

This class has the following attributes:

- operatingLocationTypeIdentifier, one or many
- operatingLocationTypeName, zero, one or many
- operatingLocationTypeDescription, zero, one or many
- numberOfOperatingLocations, zero or one

3.29.3.5.2 Associations

This class has the following associations:

- A directed has association with zero, one or many instances of MaintenanceCapabilityAtOperatingLocationType

3.29.3.5.3 Implementations

This class implements the following <<extend>> interfaces:

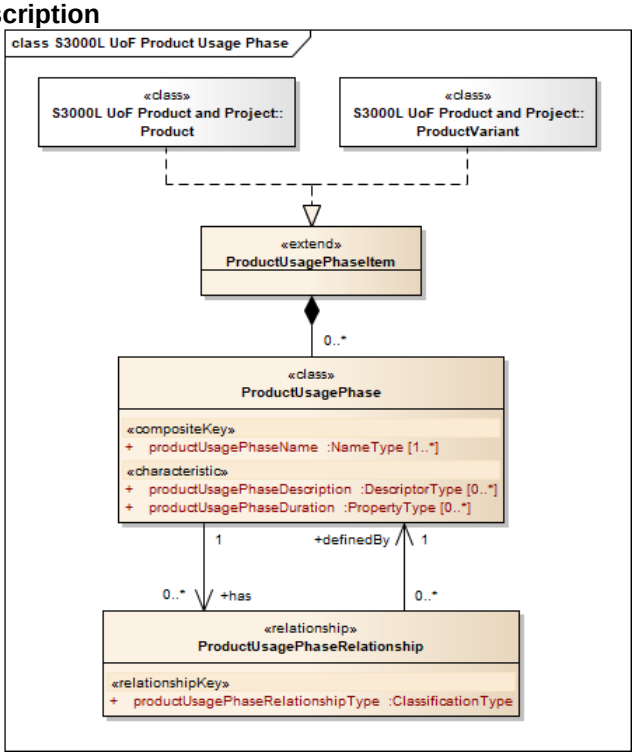
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- EnvironmentDefinitionItem. Refer to Para 3.12
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.30 S3000L UoF Product Usage Phase

3.30.1 Description

The Product Usage Phase UoF defines periods of time during which a Product is in an operational state which has specific characteristics that need special considerations.

3.30.2 Graphical description



ICN-B6865-S3000L0280-001-01

Fig 34 S3000L UoF Product Usage Phase

3.30.3 Class definition

3.30.3.1 ProductUsagePhase

`ProductUsagePhase` is a `<<class>>` that represents a distinct period of time during which a `ProductUsagePhaseItem` is in an operational state which has specific characteristics that need special considerations.

Example(s)

- Emersion, surface and dock for a submarine.
- Preflight, postflight, cruise, taxiing for an aircraft.

3.30.3.1.1 Attribute(s)

This class has the following attributes:

- `productUsagePhaseName`, one or many
- `productUsagePhaseDescription`, zero, one or many
- `productUsagePhaseDuration`, zero, one or many

3.30.3.1.2 Associations

This class has the following associations:

- A directed `has` association with zero, one or many instances of `ProductUsagePhaseRelationship`

3.30.3.1.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem. Refer to Para 3.33

3.30.3.2 ProductUsagePhaseItem

ProductUsagePhaseItem is an <<extend>> interface that provides its associated data model to those classes that implement it.

3.30.3.2.1 Class members

Classes that implement the ProductUsagePhaseItem <<extend>> interface are:

- Product. Refer to Para 3.27
- ProductVariant. Refer to Para 3.27

3.30.3.2.2 Associations

the ProductUsagePhaseItem <<extend>> interface has the following associations:

- An aggregate association with zero, one or many instances of ProductUsagePhase

3.30.3.3 ProductUsagePhaseRelationship

ProductUsagePhaseRelationship is a <<relationship>> where one ProductUsagePhase relates to another ProductUsagePhase.

3.30.3.3.1 Attribute(s)

This class has the following attributes:

- productUsagePhaseRelationshipType

3.30.3.3.2 Associations

This class has the following associations:

- A directed definedBy association with one instance of ProductUsagePhase

3.30.3.3.3 Implementations

This class implements the following <<extend>> interfaces:

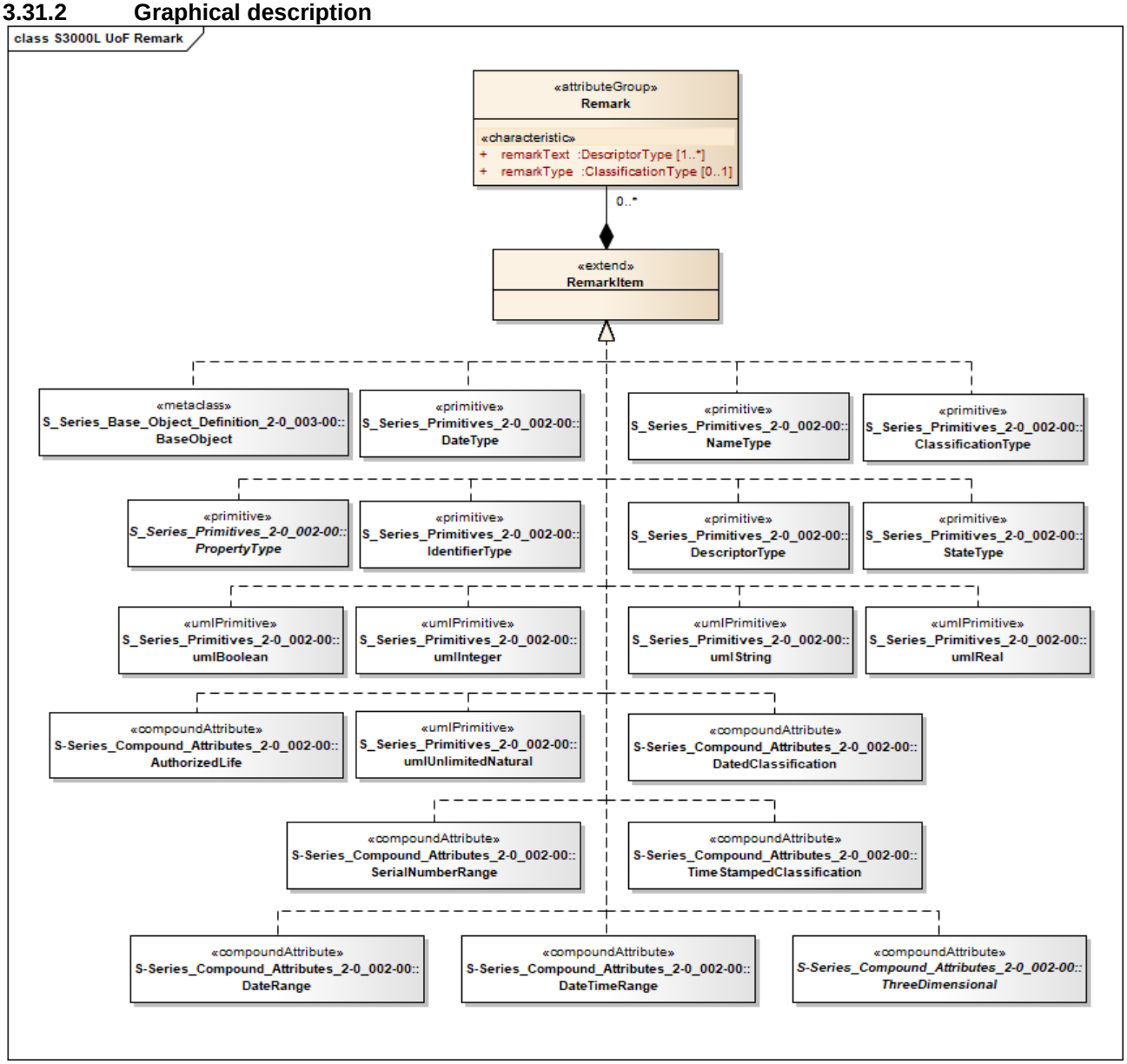
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.31 S3000L UoF Remark

3.31.1 Description

The Remark UoF provides the capability to annotate additional information relevant to the associated item which is not part of the immediate subject.

3.31.2 Graphical description



ICN-B6865-S3000L0234-004-01

Fig 35 S3000L UoF Remark

3.31.3 Class definition

3.31.3.1 Remark

`Remark` is an `<<attributeGroup>>` that provides additional information about the associated item.

Note

A remark may be a personal opinion (“I prefer more onions in my soup”) or it may be a technical fact (“The manufacturer recommends heating the soup to 45 degrees Celsius”).

3.31.3.1.1 Attribute(s)

This class has the following attributes:

- `remarkText`, one or many
- `remarkType`, zero or one

3.31.3.1.2 Implementations

This class implements the following <<extend>> interfaces:

- DigitalFileReferencingItem. Refer to Para 3.10

3.31.3.2 RemarkItem

RemarkItem is an <<extend>> interface that provides its associated data model to those classes that implement it.

3.31.3.2.1 Class members

Classes that implement the RemarkItem <<extend>> interface are:

- AllocatedTaskLocation. Refer to Para 3.39
- AllowedProductConfigurationByConfigurationIdentifier. Refer to Para 3.28
- AnalysisActivity. Refer to Para 3.21
- AnalysisActivityRevision. Refer to Para 3.21
- ApplicabilityStatement. Refer to Para 3.2
- AuthorityToOperate. Refer to Para 3.28
- AuthorizedLife. Refer to SX002D
- Breakdown. Refer to Para 3.3
- BreakdownElement. Refer to Para 3.3
- BreakdownElementRevision. Refer to Para 3.3
- BreakdownElementUsageInBreakdown. Refer to Para 3.3
- BreakdownRevision. Refer to Para 3.3
- ChangeAuthorization. Refer to Para 3.4
- ChangeRequest. Refer to Para 3.9
- CircuitBreaker. Refer to Para 3.5
- CircuitBreakerSetting. Refer to Para 3.36
- CircuitBreakerSettings. Refer to Para 3.36
- ClassificationType. Refer to SX002D
- CompetencyDefinition. Refer to Para 3.6
- ConditionInstance. Refer to Para 3.2
- ConditionType. Refer to Para 3.2
- ConditionTypeAssertMember. Refer to Para 3.2
- Contract. Refer to Para 3.27
- Country. Refer to Para 3.19
- DamageDefinition. Refer to Para 3.7
- DatedClassification. Refer to SX002D
- DateRange. Refer to SX002D
- DateTimeRange. Refer to SX002D
- DateType. Refer to SX002D
- DecisionTreeTemplate. Refer to Para 3.8
- DecisionTreeTemplateActionDefinition. Refer to Para 3.8
- DecisionTreeTemplateQuestionDefinition. Refer to Para 3.8
- DecisionTreeTemplateRevision. Refer to Para 3.8
- DescriptorType. Refer to SX002D
- DetectionMeansAlarm. Refer to Para 3.16
- DigitalFile. Refer to Para 3.10
- Document. Refer to Para 3.11

- DocumentIssue. Refer to Para 3.11
- EnvironmentDefinition. Refer to Para 3.12
- EnvironmentDefinitionCharacteristic. Refer to Para 3.12
- EnvironmentDefinitionRevision. Refer to Para 3.12
- EvaluationCriteria. Refer to Para 3.2
- Facility. Refer to Para 3.13
- FailureMode. Refer to Para 3.14
- FailureModeAnalysis. Refer to Para 3.14
- FailureModeAnalysisRevision. Refer to Para 3.14
- FailureModeCause. Refer to Para 3.14
- FailureModeEffect. Refer to Para 3.14
- FailureModeSymptomsSignature. Refer to Para 3.16
- GeographicalArea. Refer to Para 3.19
- GlobalPosition. Refer to Para 3.19
- IdentifierType. Refer to SX002D
- InServiceOptimizationAnalysis. Refer to Para 3.18
- InServiceOptimizationAnalysisRevision. Refer to Para 3.18
- InServiceOptimizationAnalysisStep. Refer to Para 3.18
- LSAFailureModeGroup. Refer to Para 3.22
- MaintenanceLevel. Refer to Para 3.29
- MeasurementPointDefinition. Refer to Para 3.16
- Message. Refer to Para 3.23
- NameType. Refer to SX002D
- OperatingLocationType. Refer to Para 3.29
- Organization. Refer to Para 3.24
- PartAsDesigned. Refer to Para 3.25
- PartAsDesignedPartsList. Refer to Para 3.25
- PartAsDesignedPartsListEntry. Refer to Para 3.25
- PartAsDesignedPartsListRevision. Refer to Para 3.25
- PerformanceParameter. Refer to Para 3.26
- PerformanceParameterRevision. Refer to Para 3.26
- Product. Refer to Para 3.27
- ProductUsagePhase. Refer to Para 3.30
- ProductVariant. Refer to Para 3.27
- Project. Refer to Para 3.27
- PropertyType. Refer to SX002D
- RepeatDefinition. Refer to Para 3.40
- ResourceSpecification. Refer to Para 3.32
- ResourceSpecificationRevision. Refer to Para 3.32
- SecurityClass. Refer to Para 3.33
- SerialNumberRange. Refer to SX002D
- Skill. Refer to Para 3.6
- SkillLevel. Refer to Para 3.6
- SpecialEventDefinition. Refer to Para 3.35
- StateType. Refer to SX002D
- StreetAddress. Refer to Para 3.19
- SubstanceDefinition. Refer to Para 3.25
- Subtask. Refer to Para 3.36

- Task. Refer to Para 3.36
- TaskRequirement. Refer to Para 3.37
- TaskRequirementRevision. Refer to Para 3.37
- TaskResource. Refer to Para 3.38
- TaskRevision. Refer to Para 3.36
- ThreeDimensional. Refer to SX002D
- ThresholdDefinition. Refer to Para 3.40
- TimeLimit. Refer to Para 3.40
- TimeStampedClassification. Refer to SX002D
- Trade. Refer to Para 3.6
- umlBoolean. Refer to SX002D
- umlInteger. Refer to SX002D
- umlReal. Refer to SX002D
- umlString. Refer to SX002D
- umlUnlimitedNatural. Refer to SX002D
- WarningCautionNote. Refer to Para 3.36

3.31.3.2.2 Associations

The RemarkItem <<extend>> interface has the following associations:

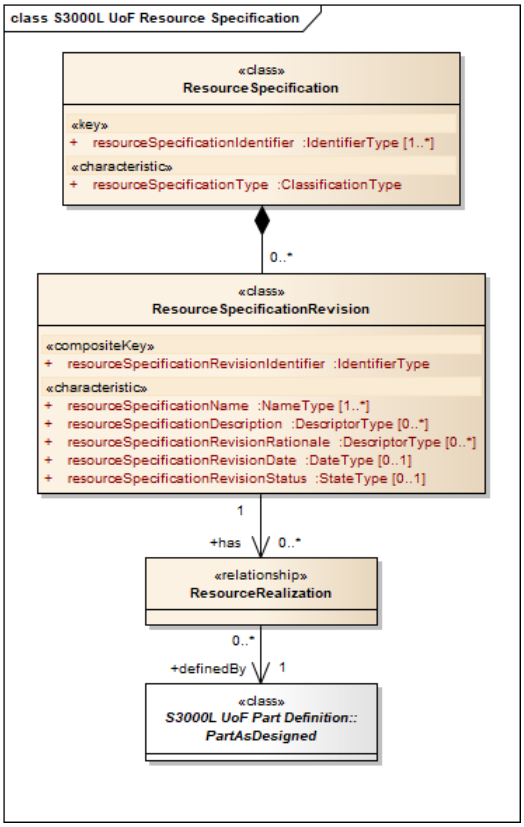
- An aggregate association with zero, one or many instances of Remark

3.32 S3000L UoF Resource Specification

3.32.1 Description

The Resource Specification UoF provides the capability to define generic definitions of resources needed without having to define its actual realization.

3.32.2 Graphical description



ICN-B6865-S3000L0285-001-01

Fig 36 S3000L UoF Resource Specification

3.32.3 Class definition

3.32.3.1 ResourceRealization

`ResourceRealization` is a `<<relationship>>` where a `ResourceSpecification` relates to an instance of `PartAsDesigned` (refer to Para 3.25) that fulfills the resource specification.

3.32.3.1.1 Associations

This class has the following associations:

- A directed `definedBy` association with one instance of `PartAsDesigned`. Refer to Para 3.25

3.32.3.1.2 Implementations

This class implements the following `<<extend>>` interfaces:

- `ApplicabilityStatementItem`. Refer to Para 3.2
- `ChangeControlledItem`. Refer to Para 3.4
- `DocumentReferencingItem` (inherited from `BaseObject`). Refer to Para 3.11
- `OrganizationReferencingItem` (inherited from `BaseObject`). Refer to Para 3.24
- `ProjectSpecificExtensionItem` (inherited from `BaseObject`). Refer to SX002D
- `RemarkItem` (inherited from `BaseObject`). Refer to Para 3.31

3.32.3.2 ResourceSpecification

ResourceSpecification is a <<class>> that defines a resource by its characteristics.

Note

ResourceSpecification allows for more generic resource definitions ie, a task/subtask does not need to be changed due to eg, customer specific resource preferences.

3.32.3.2.1 Attribute(s)

This class has the following attributes:

- resourceSpecificationIdentifier, one or many
- resourceSpecificationType

3.32.3.2.2 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of ResourceSpecificationRevision

3.32.3.2.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem. Refer to Para 3.33

3.32.3.3 ResourceSpecificationRevision

ResourceSpecificationRevision is a <<class>> representing an iteration applied to a ResourceSpecification.

3.32.3.3.1 Attribute(s)

This class has the following attributes:

- resourceSpecificationRevisionIdentifier
- resourceSpecificationName, one or many
- resourceSpecificationDescription, zero, one or many
- resourceSpecificationRevisionRationale, zero, one or many
- resourceSpecificationRevisionDate, zero or one
- resourceSpecificationRevisionStatus, zero or one

3.32.3.3.2 Associations

This class has the following associations:

- A directed has association with zero, one or many instances of ResourceRealization

3.32.3.3.3 Implementations

This class implements the following <<extend>> interfaces:

- ChangeControlledItem. Refer to Para 3.4
- DigitalFileReferencingItem. Refer to Para 3.10

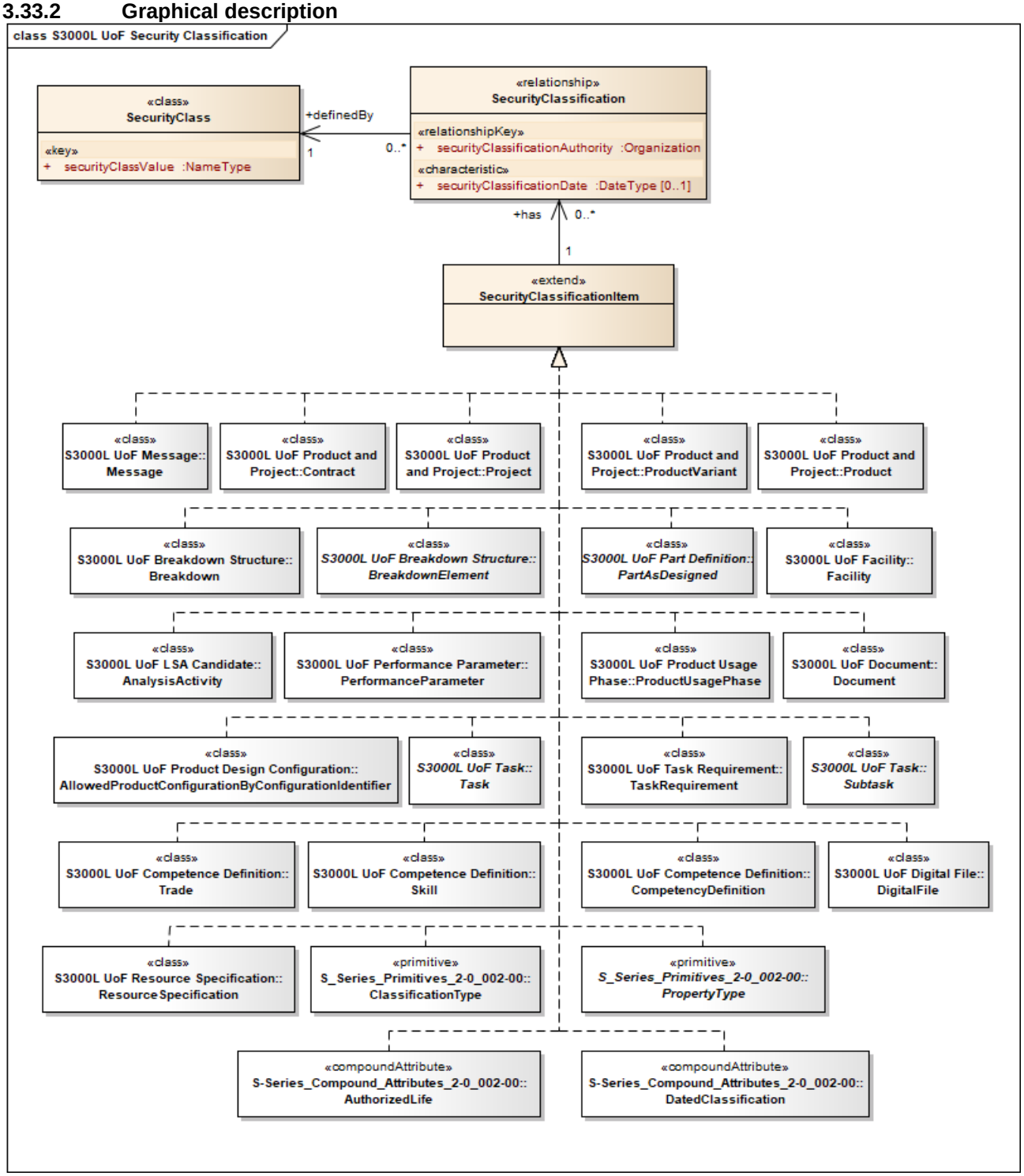
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.33 S3000L UoF Security Classification

3.33.1 Description

The Security Classification UoF provides the capability to assign security classifications to objects that need special handling for protection against unauthorized access or distribution.

3.33.2 Graphical description



ICN-B6865-S3000L0231-004-01

Fig 37 S3000L UoF Security Classification

3.33.3 Class definition

3.33.3.1 SecurityClass

`SecurityClass` is a `<<class>>` that identifies a level of confidentiality which can be used to protect something against unauthorized access.

3.33.3.1.1 Attribute(s)
This class has the following attributes:

- securityClassValue

3.33.3.1.2 Implementations
This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.33.3.2 SecurityClassification
SecurityClassification is a <<relationship>> that associates a given SecurityClass with the item that must be protected against unauthorized access or distribution

3.33.3.2.1 Attribute(s)
This class has the following attributes:

- securityClassificationAuthority
- securityClassificationDate, zero or one

3.33.3.2.2 Associations
This class has the following associations:

- A directed definedBy association with one instance of SecurityClass

3.33.3.2.3 Implementations
This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- DigitalFileReferencingItem. Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.33.3.3 SecurityClassificationItem
SecurityClassificationItem is an <<extend>> interface that provides its associated data model to those classes that implement it.

3.33.3.3.1 Class members
Classes that implement the SecurityClassificationItem <<extend>> interface are:

- AllowedProductConfigurationByConfigurationIdentifier. Refer to Para 3.28
- AnalysisActivity. Refer to Para 3.21
- AuthorizedLife. Refer to SX004G
- Breakdown. Refer to Para 3.3

- BreakdownElement. Refer to Para 3.3
- ClassificationType. Refer to SX004G
- CompetencyDefinition. Refer to Para 3.6
- Contract. Refer to Para 3.27
- DatedClassification. Refer to SX004G
- DigitalFile. Refer to Para 3.10
- Document. Refer to Para 3.11
- Facility. Refer to Para 3.13
- Message. Refer to Para 3.23
- PartAsDesigned. Refer to Para 3.25
- PerformanceParameter
- Product. Refer to Para 3.27
- ProductUsagePhase. Refer to Para 3.30
- ProductVariant. Refer to Para 3.27
- Project. Refer to Para 3.27
- PropertyType. Refer to SX004G
- ResourceSpecification. Refer to Para 3.32
- Skill. Refer to Para 3.6
- Subtask. Refer to Para 3.36
- Task. Refer to Para 3.36
- TaskRequirement. Refer to Para 3.37
- Trade. Refer to Para 3.6

3.33.3.3.2 Associations

The SecurityClassificationItem <<extend>> interface has the following associations:

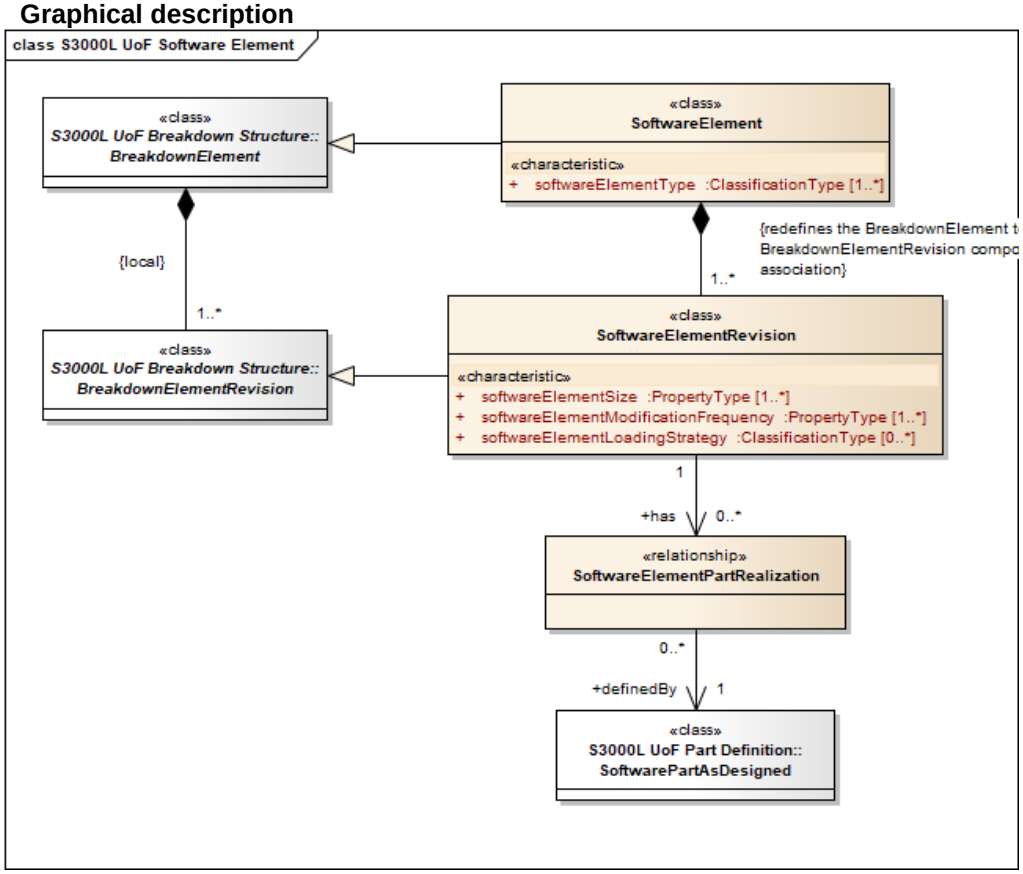
- A directed has association with zero, one or many instances of SecurityClassification

3.34 S3000L UoF Software Element

3.34.1 Description

The Software Element UoF provides the capability to specify that an element within a breakdown is software and can be associated with the software part(s) that fulfill the requirement.

3.34.2 Graphical description



ICN-B6865-S3000L0274-001-01

Fig 38 S3000L UoF Software Element

3.34.3 Class definition

3.34.3.1 SoftwareElement

`SoftwareElement` is a `BreakdownElement` (refer to Para 3.3) that is realized as a `SoftwarePartAsDesigned` (refer to Para 3.25).

3.34.3.1.1 Attribute(s)

This class has the following attributes:

- `breakdownElementIdentifier` (inherited from `BreakdownElement`), one or many
- `breakdownElementName` (inherited from `BreakdownElement`), zero, one or many
- `breakdownElementEssentiality` (inherited from `BreakdownElement`), zero or one
- `softwareElementType`, one or many

3.34.3.1.2 Associations

This class has the following associations:

- An aggregate association with one or many instances of `SoftwareElementRevision`

3.34.3.1.3 Implementations

This class implements the following `<<extend>>` interfaces:

- AnalysisCandidateItem (inherited from BreakdownElement). Refer to Para 3.21
- BreakdownElementInZoneItem (inherited from BreakdownElement). Refer to Para 3.41
- DecisionTreeAnalysisItem (inherited from BreakdownElement). Refer to Para 3.8
- DetectionMeansAlarmItem (inherited from BreakdownElement). Refer to Para 3.16
- DigitalFileReferencingItem (inherited from BreakdownElement). Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- FailureModeAnalysisItem (inherited from BreakdownElement). Refer to Para 3.14
- InServiceOptimizationAnalysisItem (inherited from BreakdownElement). Refer to Para 3.18
- MeasurementPointDefinitionItem (inherited from BreakdownElement). Refer to Para 3.16
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- PerformanceParameterItem (inherited from BreakdownElement). Refer to Para 3.26
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem (inherited from BreakdownElement). Refer to Para 3.33

3.34.3.2 SoftwareElementPartRealization

SoftwareElementPartRealization is a <<relationship>> where a SoftwareElementRevision relates to an instance of SoftwarePartAsDesigned (refer to Para 3.25) which fulfills the SoftwareElement specification.

3.34.3.2.1 Associations

This class has the following associations:

- A directed definedBy association with one instance of SoftwarePartAsDesigned. Refer to Para 3.25

3.34.3.2.2 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- ChangeControlledItem. Refer to Para 3.4
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- EffectiveOnProductConfigurationItem. Refer to Para 3.28
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- UsableOnItem. Refer to Para 3.28

3.34.3.3 SoftwareElementRevision

SoftwareElementRevision is a BreakdownElementRevision (refer to Para 3.3) representing an iteration applied to a SoftwareElement.

3.34.3.3.1 Attribute(s)

This class has the following attributes:

- breakdownElementRevisionIdentifier (inherited from BreakdownElementRevision)
- breakdownElementDescription (inherited from BreakdownElementRevision), zero, one or many
- maintenanceSignificantOrRelevant (inherited from BreakdownElementRevision)
- breakdownElementRevisionRationale (inherited from BreakdownElementRevision), zero, one or many
- breakdownElementRevisionDate (inherited from BreakdownElementRevision), zero or one
- breakdownElementRevisionStatus (inherited from BreakdownElementRevision), zero or one
- softwareElementSize, one or many
- softwareElementModificationFrequency, one or many
- softwareElementLoadingStrategy, zero, one or many

3.34.3.3.2 Associations

This class has the following associations:

- A directed has association with zero, one or many instances of BreakdownElementRevisionRelationship (inherited from BreakdownElementRevision). Refer to Para 3.3
- A directed has association with zero, one or many instances of SoftwareElementPartRealization

3.34.3.3.3 Implementations

This class implements the following <<extend>> interfaces:

- AnalysisCandidateItem (inherited from BreakdownElementRevision). Refer to Para 3.21
- BreakdownElementInZoneItem (inherited from BreakdownElementRevision). Refer to Para 3.41
- ChangeControlledItem (inherited from BreakdownElementRevision). Refer to Para 3.4
- DetectionMeansAlarmItem (inherited from BreakdownElementRevision). Refer to Para 3.16
- DigitalFileReferencingItem (inherited from BreakdownElementRevision). Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- FailureModeAnalysisItem (inherited from BreakdownElementRevision). Refer to Para 3.14
- InServiceOptimizationAnalysisItem (inherited from BreakdownElementRevision). Refer to Para 3.18
- MeasurementPointDefinitionItem (inherited from BreakdownElementRevision). Refer to Para 3.16

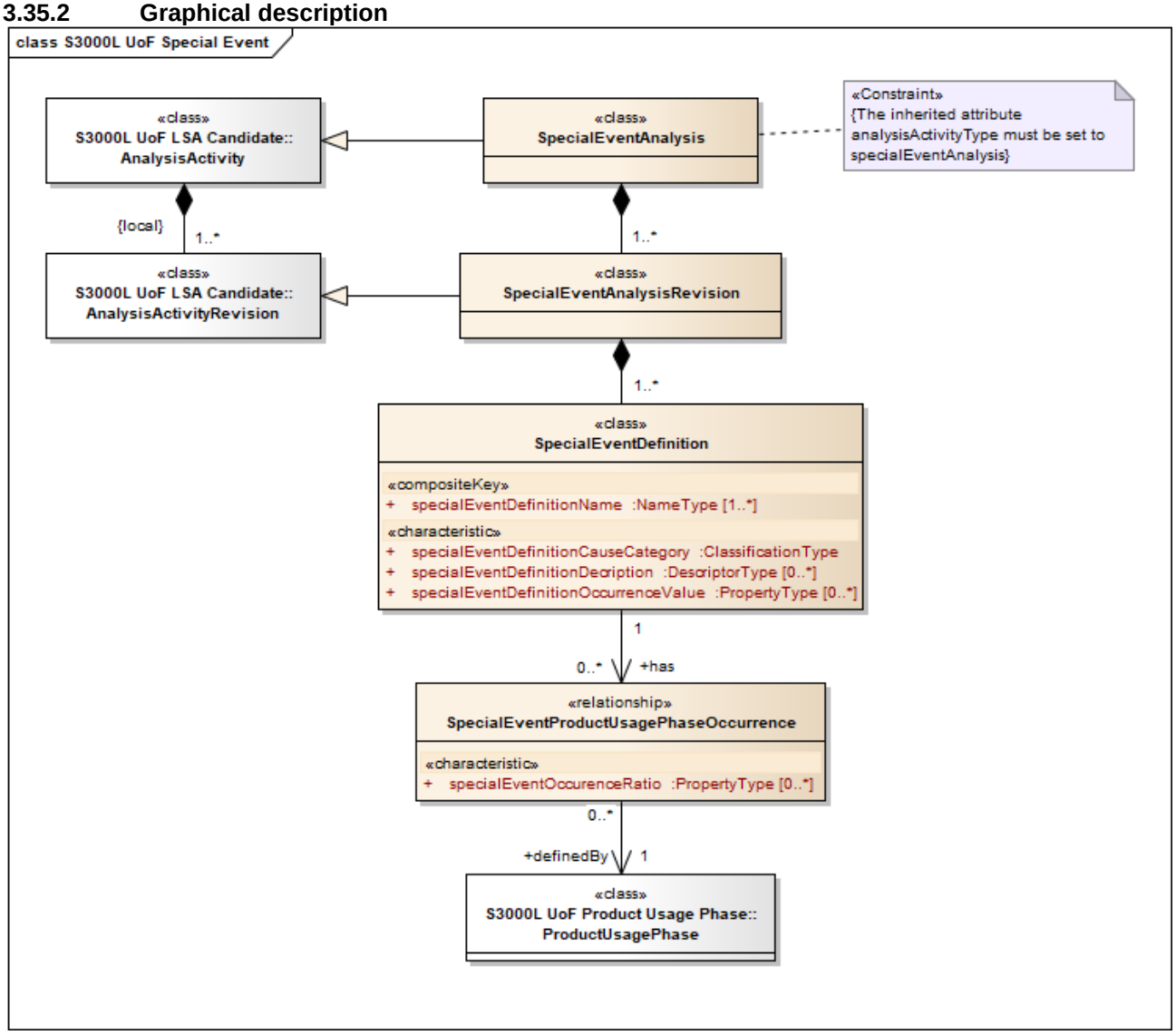
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- PerformanceParameterItem (inherited from BreakdownElementRevision). Refer to Para 3.26
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- TaskAnalysisItem (inherited from BreakdownElementRevision). Refer to Para 3.39
- TaskRequirementAnalysisItem (inherited from BreakdownElementRevision). Refer to Para 3.37

3.35 S3000L UoF Special Event

3.35.1 Description

The Special Event UoF provides the capability to define typical happenings which can be induced to an item under analysis and which can impact the operational capability of the Product during its in-service phase.

3.35.2 Graphical description



ICN-B6865-S3000L0281-001-01

Fig 39 S3000L UoF Special Event

3.35.3 Class definition

3.35.3.1 SpecialEventAnalysis

`SpecialEventAnalysis` is `AnalysisActivity` (refer to Para 3.21) that represents the objective for, and outcome of, a special event analysis carried out for the `AnalysisCandidateItem` (refer to Para 3.21).

3.35.3.1.1 Attribute(s)

This class has the following attributes:

- `analysisActivityType` (inherited from `AnalysisActivity`)

3.35.3.1.2 Associations

This class has the following associations:

- An aggregate association with one or many instances of `SpecialEventAnalysisRevision`

3.35.3.1.3 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem (inherited from AnalysisActivity). Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem (inherited from AnalysisActivity). Refer to Para 3.33

3.35.3.2 SpecialEventAnalysisRevision

SpecialEventAnalysisRevision is an AnalysisActivityRevision (refer to Para 3.21) representing an iteration applied to a SpecialEventAnalysis.

3.35.3.2.1 Attribute(s)

This class has the following attributes:

- analysisActivityRevisionIdentifier (inherited from AnalysisActivityRevision)
- analysisActivityDecision (inherited from AnalysisActivityRevision)
- analysisActivityDecisionRationale (inherited from AnalysisActivityRevision), zero, one or many
- analysisActivityStatusDescription (inherited from AnalysisActivityRevision), zero, one or many
- analysisActivityRevisionRationale (inherited from AnalysisActivityRevision), zero, one or many
- analysisActivityRevisionDate (inherited from AnalysisActivityRevision), zero or one
- analysisActivityRevisionStatus (inherited from AnalysisActivityRevision), zero or one

3.35.3.2.2 Associations

This class has the following associations:

- An aggregate association with one or many instances of SpecialEventDefinition

3.35.3.2.3 Implementations

This class implements the following <<extend>> interfaces:

- ChangeControlledItem (inherited from AnalysisActivityRevision). Refer to Para 3.4
- DigitalFileReferencingItem (inherited from AnalysisActivityRevision). Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24

- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.35.3.3 SpecialEventDefinition

SpecialEventDefinition is a <<class>>that represents a typical happening which can be induced to the item under analysis during its normal operation, and can lead to damages.

3.35.3.3.1 Attribute(s)

This class has the following attributes:

- specialEventDefinitionName, one or many
- specialEventDefinitionCauseCategory
- specialEventDefinitionDecription, zero, one or many
- specialEventDefinitionOccurrenceValue, zero, one or many

3.35.3.3.2 Associations

This class has the following associations:

- A directed has association with zero, one or many instances of SpecialEventProductUsagePhaseOccurrence

3.35.3.3.3 Implementations

This class implements the following <<extend>> interfaces:

- DigitalFileReferencingItem. Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.35.3.4 SpecialEventProductUsagePhaseOccurrence

SpecialEventProductUsagePhaseOccurrence is a <<relationship>> that defines an association between an instance of SpecialEventDefinition and an instance of ProductUsagePhase (refer to Para 3.30) during which the special event can occur.

3.35.3.4.1 Attribute(s)

This class has the following attributes:

- specialEventOccurenceRatio, zero, one or many

3.35.3.4.2 Associations

This class has the following associations:

- A directed definedBy association with one instance of ProductUsagePhase. Refer to Para 3.30

3.35.3.4.3 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11

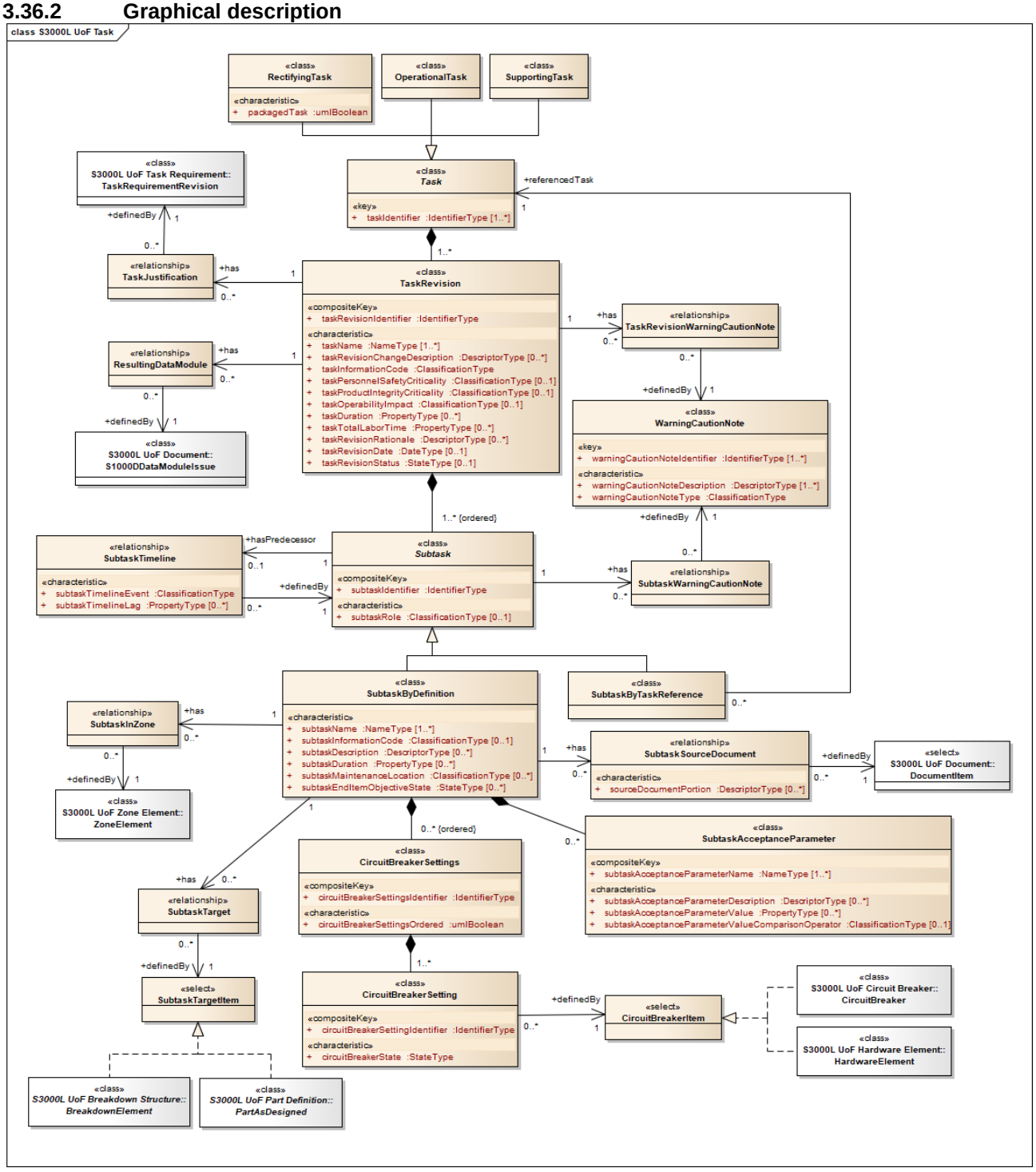
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.36 S3000L UoF Task

3.36.1 Description

The Task UoF supports the detailed definition of tasks required to support a Product.

3.36.2 Graphical description



ICN-B6865-S3000L0227-004-01

Fig 40 S3000L UoF Task

3.36.3 Class definition

3.36.3.1 CircuitBreakerItem

`CircuitBreakerItem` is a `<<select>>` interface that identifies items which can represent the referenced circuit breaker.

3.36.3.1.1 Class members

This <<select>> interface includes the following class members:

- CircuitBreaker. Refer to Para 3.5
- HardwareElement. Refer to Para 3.17

3.36.3.2 CircuitBreakerSetting

CircuitBreakerSetting is a <<class>> that specifies an individual circuit breaker that must be in a specific state.

3.36.3.2.1 Attribute(s)

This class has the following attributes:

- circuitBreakerSettingIdentifier
- circuitBreakerState

3.36.3.2.2 Associations

This class has the following associations:

- A directed definedBy association with one instance of CircuitBreakerItem

3.36.3.2.3 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.36.3.3 CircuitBreakerSettings

CircuitBreakerSettings is a <<class>> that identifies a set of circuit breakers that must be set in specific states.

3.36.3.3.1 Attribute(s)

This class has the following attributes:

- circuitBreakerSettingsIdentifier
- circuitBreakerSettingsOrdered

3.36.3.3.2 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of CircuitBreakerSetting

3.36.3.3.3 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D

- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.36.3.4 OperationalTask

OperationalTask is a Task that is required to support the use of a product.

Example(s)
- Fueling
- Towing

3.36.3.4.1 Attribute(s)

This class has the following attributes:

- taskIdentifier (inherited from Task), one or many

3.36.3.4.2 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of TaskRevision (inherited from Task).

3.36.3.4.3 Implementations

This class implements the following <<extend>> interfaces:

- DigitalFileReferencingItem (inherited from Task). Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- EnvironmentDefinitionItem (inherited from Task). Refer to Para 3.12
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem (inherited from Task). Refer to Para 3.33

3.36.3.5 RectifyingTask

RectifyingTask is a Task that ensures or returns the function of the associated item.

Note

An event that can require a Task to be performed includes a failure, damage, a special event, and a time limit.

Example(s)
- Lubrication
- Repair
- Replace

3.36.3.5.1 Attribute(s)

This class has the following attributes:

- taskIdentifier (inherited from Task), one or many
- packagedTask

3.36.3.5.2 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of TaskRevision (inherited from Task).

3.36.3.5.3 Implementations
This class implements the following <<extend>> interfaces:

- DigitalFileReferencingItem (inherited from Task). Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- EnvironmentDefinitionItem (inherited from Task). Refer to Para 3.12
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem (inherited from Task). Para 3.33

3.36.3.6 ResultingDataModule
ResultingDataModule is a <<relationship>> that identifies a data module issue in which a TaskRevision is further detailed.

Note
A data module can contain either a complete or partial description of a given task revision.

3.36.3.6.1 Associations
This class has the following associations:

- A directed definedBy association with one instance of S1000DDataModuleIssue. Refer to Para 3.11

3.36.3.6.2 Implementations
This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.36.3.7 Subtask
Subtask is a <<class>> that represents the specification of a work step that is to be performed as part of a Task.

3.36.3.7.1 Attribute(s)
This class has the following attributes:

- subtaskIdentifier
- subtaskRole, zero or one

3.36.3.7.2 Associations
This class has the following associations:

- A directed hasPredecessor association with zero or one instance of SubtaskTimeline
- A directed has association with zero, one or many instances of SubtaskWarningCautionNote

3.36.3.7.3 Implementations
This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- DigitalFileReferencingItem. Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem. Refer to Para 3.33

3.36.3.8 SubtaskAcceptanceParameter

SubtaskAcceptanceParameter is a <<class>> that represents acceptance criteria which must be met before the Subtask is completed.

3.36.3.8.1 Attribute(s)

This class has the following attributes:

- subtaskAcceptanceParameterName, one or many
- subtaskAcceptanceParameterDescription, zero, one or many
- subtaskAcceptanceParameterValue, zero, one or many
- subtaskAcceptanceParameterValueComparisonOperator, zero or one

3.36.3.8.2 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.36.3.9 SubtaskByDefinition

SubtaskByDefinition is a Subtask that provides detailed information of the defined work step.

3.36.3.9.1 Attribute(s)

This class has the following attributes:

- subtaskIdentifier (inherited from Subtask)
- subtaskRole (inherited from Subtask), zero or one
- subtaskName, one or many
- subtaskInformationCode, zero or one
- subtaskDescription, zero, one or many
- subtaskDuration, zero, one or many
- subtaskMaintenanceLocation, zero, one or many
- subtaskEndItemObjectiveState, zero, one or many

3.36.3.9.2 Associations

This class has the following associations:

- An ordered aggregate association with zero, one or many instances of CircuitBreakerSettings
- An aggregate association with zero, one or many instances of SubtaskAcceptanceParameter
- A directed hasPredecessor association with zero or one instance of SubtaskTimeline (inherited from Subtask)
- A directed has association with zero, one or many instances of SubtaskWarningCautionNote (inherited from Subtask)
- A directed has association with zero, one or many instances of SubtaskInZone
- A directed has association with zero, one or many instances of SubtaskSourceDocument
- A directed has association with zero, one or many instances of SubtaskTarget

3.36.3.9.3 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem (inherited from Subtask). Refer to Para 3.2
- DigitalFileReferencingItem (inherited from Subtask). Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem (inherited from Subtask). Refer to Para 3.33
- TaskResourceItem. Refer to Para 3.38

3.36.3.10 SubtaskByTaskReference

SubtaskByTaskReference is a Subtask where the details of the subtask are defined as a separate Task.

3.36.3.10.1 Attribute(s)

This class has the following attributes:

- subtaskIdentifier (inherited from Subtask)
- subtaskRole (inherited from Subtask), zero or one

3.36.3.10.2 Associations

This class has the following associations:

- A directed hasPredecessor association with zero or one instance of SubtaskTimeline (inherited from Subtask)
- A directed has association with zero, one or many instances of SubtaskWarningCautionNote (inherited from Subtask)
- A directed referencedTask association with one instance of Task

3.36.3.10.3 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem (inherited from Subtask). Refer to Para 3.2
- DigitalFileReferencingItem (inherited from Subtask). Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11

- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem (inherited from Subtask). Refer to Para 3.33

3.36.3.11 SubtaskInZone

SubtaskInZone is a <<relationship>> that identifies the zone where the Subtask is to be performed.

3.36.3.11.1 Associations

This class has the following associations:

- A directed definedBy association with one instance of ZoneElement. Refer to Para 3.41

3.36.3.11.2 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.36.3.12 SubtaskSourceDocument

SubtaskSourceDocument is a <<relationship>> that identifies an external Document (refer to Para 3.11) where the Subtask is originally defined and more details are given.

3.36.3.12.1 Attribute(s)

This class has the following attributes:

- sourceDocumentPortion, zero, one or many

3.36.3.12.2 Associations

This class has the following associations:

- A directed definedBy association with one instance of DocumentItem. Refer to Para 3.11

3.36.3.12.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.36.3.13 SubtaskTarget

SubtaskTarget is a <<relationship>> that identifies the item on which the Subtask is to be performed.

3.36.3.13.1 Associations

This class has the following associations:

- A directed definedBy association one instance of SubtaskTargetItem

3.36.3.13.2 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.36.3.14 SubtaskTargetItem

SubtaskTargetItem is a <<select>> interface that identifies items on which a Subtask can be performed.

3.36.3.14.1 Class members

This <<select>> interface includes the following class members:

- BreakdownElement. Refer to Para 3.3
- PartAsDesigned. Refer to Para 3.25

3.36.3.15 SubtaskTimeline

SubtaskTimeline is a <<relationship>> that identifies that there is a time dependency between two Subtasks within the same Task.

3.36.3.15.1 Attribute(s)

This class has the following attributes:

- subtaskTimelineEvent
- subtaskTimelineLag, zero, one or many

3.36.3.15.2 Associations

This class has the following associations:

- A directed definedBy association with one instance of Subtask

3.36.3.15.3 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.36.3.16 SubtaskWarningCautionNote

SubtaskWarningCautionNote is a <<relationship>> that identifies a WarningCautionNote that is associated with a given Subtask.

3.36.3.16.1 Associations

This class has the following associations:

- A directed definedBy association with one instance of WarningCautionNote

3.36.3.16.2 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.36.3.17 SupportingTask

SupportingTask is a Task that does not meet a TaskRequirement, but identifies a set of work steps which will be carried out as part of multiple Tasks.

Note

The objective for a SupportingTask is to enable reuse of a sequence of work steps, needed by a set of Tasks.

Note

A SupportingTask will only be used in the context of SubtaskByTaskReference.

Example(s)

- Jack vehicle
- Open hatch

3.36.3.17.1 Attribute(s)

This class has the following attributes:

- taskIdentifier (inherited from Task), one or many

3.36.3.17.2 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of TaskRevision (inherited from Task).

3.36.3.17.3 Implementations

This class implements the following <<extend>> interfaces:

- DigitalFileReferencingItem (inherited from Task). Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- EnvironmentDefinitionItem (inherited from Task). Refer to Para 3.12
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24

- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem (inherited from Task). Refer to Para 3.33

3.36.3.18 Task

Task is a <<class>> that represents the specification of work to be done or undertaken.

3.36.3.18.1 Attribute(s)

This class has the following attributes:

- taskIdentifier, one or many

3.36.3.18.2 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of TaskRevision.

3.36.3.18.3 Implementations

This class implements the following <<extend>> interfaces:

- DigitalFileReferencingItem. Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- EnvironmentDefinitionItem. Refer to Para 3.12
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem. Refer to Para 3.33

3.36.3.19 TaskJustification

TaskJustification is a <<relationship>> that identifies a TaskRequirement (refer to Para 3.37) that defines the need for the existence of a Task.

3.36.3.19.1 Associations

This class has the following associations:

- A directed definedBy association with one instance of TaskRequirementRevision. Refer to Para 3.37

3.36.3.19.2 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.36.3.20 TaskRevision

TaskRevision is a <<class>> representing an iteration applied to a Task.

3.36.3.20.1 Attribute(s)

This class has the following attributes:

- taskRevisionIdentifier
- taskName, one or many
- taskRevisionChangeDescription, zero, one or many
- taskInformationCode
- taskPersonnelSafetyCriticality, zero or one
- taskProductIntegrityCriticality, zero or one
- taskOperabilityImpact, zero or one
- taskDuration, zero, one or many
- taskTotalLaborTime, zero, one or many
- taskRevisionRationale, zero, one or many
- taskRevisionDate, zero or one
- taskRevisionStatus, zero or one

3.36.3.20.2 Associations

This class has the following associations:

- An ordered aggregate association with one or many instances of Subtask
- A directed has association with zero, one or many instances of ResultingDataModule
- A directed has association with zero, one or many instances of TaskJustification
- A directed has association with zero, one or many instances of TaskRevisionWarningCautionNote

3.36.3.20.3 Implementations

This class implements the following <<extend>> interfaces:

- ChangeControlledItem. Refer to Para 3.4
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- EnvironmentDefinitionItem. Refer to Para 3.12
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- TaskResourceItem. Refer to Para 3.38

3.36.3.21 TaskRevisionWarningCautionNote

TaskRevisionWarningCautionNote is a <<relationship>> that identifies a WarningCautionNote that is associated with a given Task.

3.36.3.21.1 Associations

This class has the following associations:

- A directed definedBy association with one instance of WarningCautionNote

3.36.3.21.2 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24

- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.36.3.22 WarningCautionNote

WarningCautionNote is a <<class>> that defines advice concerning safety, legal and health aspects.

3.36.3.22.1 Attribute(s)

This class has the following attributes:

- warningCautionNoteIdentifier, one or many
- warningCautionNoteDescription, one or many
- warningCautionNoteType

3.36.3.22.2 Implementations

This class implements the following <<extend>> interfaces:

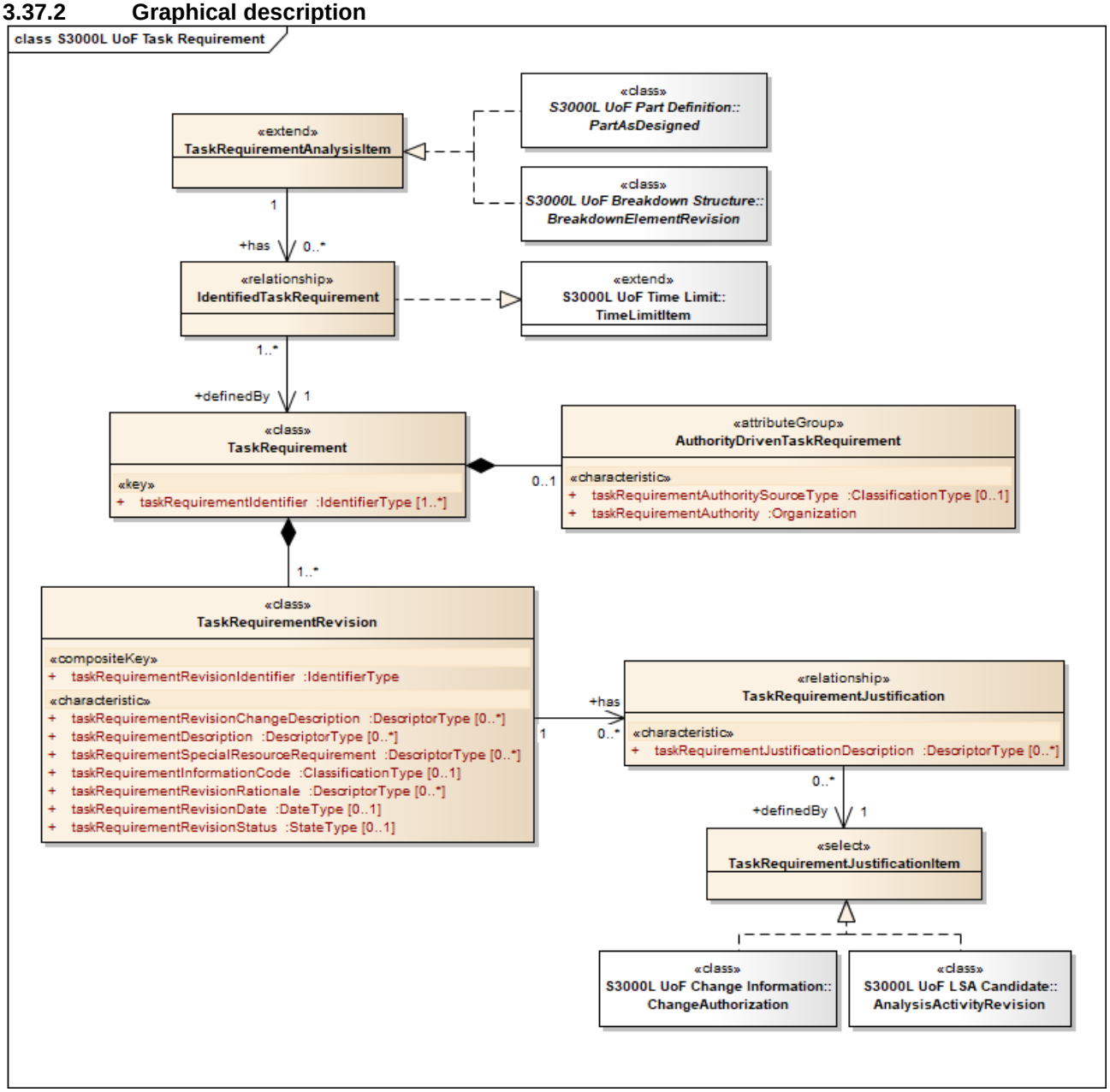
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.37 S3000L UoF Task Requirement

3.37.1 Description

The Task Requirement UoF supports early documentation of the need for a task to be performed to support a Product.

3.37.2 Graphical description



ICN-B6865-S3000L0226-004-01

Fig 41 S3000L UoF Task Requirement

3.37.3 Class definition

3.37.3.1 AuthorityDrivenTaskRequirement

`AuthorityDrivenTaskRequirement` is an `<<attributeGroup>>` that collects information on task requirement derived from regulations and/or other authoritative sources.

3.37.3.1.1 Attribute(s)

This `<<attributeGroup>>` has the following attributes:

- `taskRequirementAuthoritySourceType`, zero or one
- `taskRequirementAuthority`

3.37.3.2 IdentifiedTaskRequirement

IdentifiedTaskRequirement is a <<relationship>> that associates a TaskRequirement with a TaskRequirementAnalysisItem.

3.37.3.2.1 Associations

This class has the following associations:

- A directed definedBy association with one instance of TaskRequirement

3.37.3.2.2 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- TimeLimitItem. Refer to Para 3.40

3.37.3.3 TaskRequirement

TaskRequirement is a <<class>> that represents the need for a procedure to be developed and documented.

Note

Task requirements are identified and documented prior to any detailed task analysis.

Note

A task requirement can have more than one task being developed for different usage scenarios.

Note

Examples of support analysis activities which result in a set of documented task requirements are: Preventive maintenance analysis (refer to S4000P) and special event analysis.

3.37.3.3.1 Attribute(s)

This class has the following attributes:

- taskRequirementIdentifier, one or many

3.37.3.3.2 Associations

This class has the following associations:

- An aggregate association with zero or one instance of AuthorityDrivenTaskRequirement
- An aggregate association with one or many instances of TaskRequirementRevision

3.37.3.3.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D

- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem. Refer to Para 3.33

3.37.3.4 TaskRequirementAnalysisItem

TaskRequirementAnalysisItem is an <<extend>> interface that provides its associated data model to those classes that implement it.

3.37.3.4.1 Class members

Classes that implement the TaskRequirementAnalysisItem <<extend>> interface are:

- BreakdownElementRevision. Refer to Para 3.3
- PartAsDesigned. Refer to Para 3.25

3.37.3.4.2 Associations

The TaskRequirementAnalysisItem <<extend>> interface has the following associations:

- A directed has association with zero, one or many instances of IdentifiedTaskRequirement

3.37.3.5 TaskRequirementJustification

TaskRequirementJustification is a <<relationship>> that identifies a source which defines the need for a task.

3.37.3.5.1 Attribute(s)

This class has the following attributes:

- taskRequirementJustificationDescription, zero, one or many

3.37.3.5.2 Associations

This class has the following associations:

- A directed definedBy association with one instance of TaskRequirementJustificationItem

3.37.3.5.3 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.37.3.6 TaskRequirementJustificationItem

TaskRequirementJustificationItem is a <<select>> interface that identifies items which can be selected as being the source that justifies the TaskRequirement.

3.37.3.6.1 Class members

This <<select>> interface includes the following class members:

- AnalysisActivityRevision. Refer to Para 3.21
- ChangeAuthorization. Refer to Para 3.4

3.37.3.7 TaskRequirementRevision
TaskRequirementRevision is a <<class>> representing an iteration applied to a TaskRequirement.

3.37.3.7.1 Attribute(s)
This class has the following attributes:

- taskRequirementRevisionIdentifier
- taskRequirementRevisionChangeDescription, zero, one or many
- taskRequirementDescription, zero, one or many
- taskRequirementSpecialResourceRequirement, zero, one or many
- taskRequirementInformationCode, zero or one
- taskRequirementRevisionRationale, zero, one or many
- taskRequirementRevisionDate, zero or one
- taskRequirementRevisionStatus, zero or one

3.37.3.7.2 Associations
This class has the following associations:

- A directed has association with zero, one or many instances of TaskRequirementJustification

3.37.3.7.3 Implementations
This class implements the following <<extend>> interfaces:

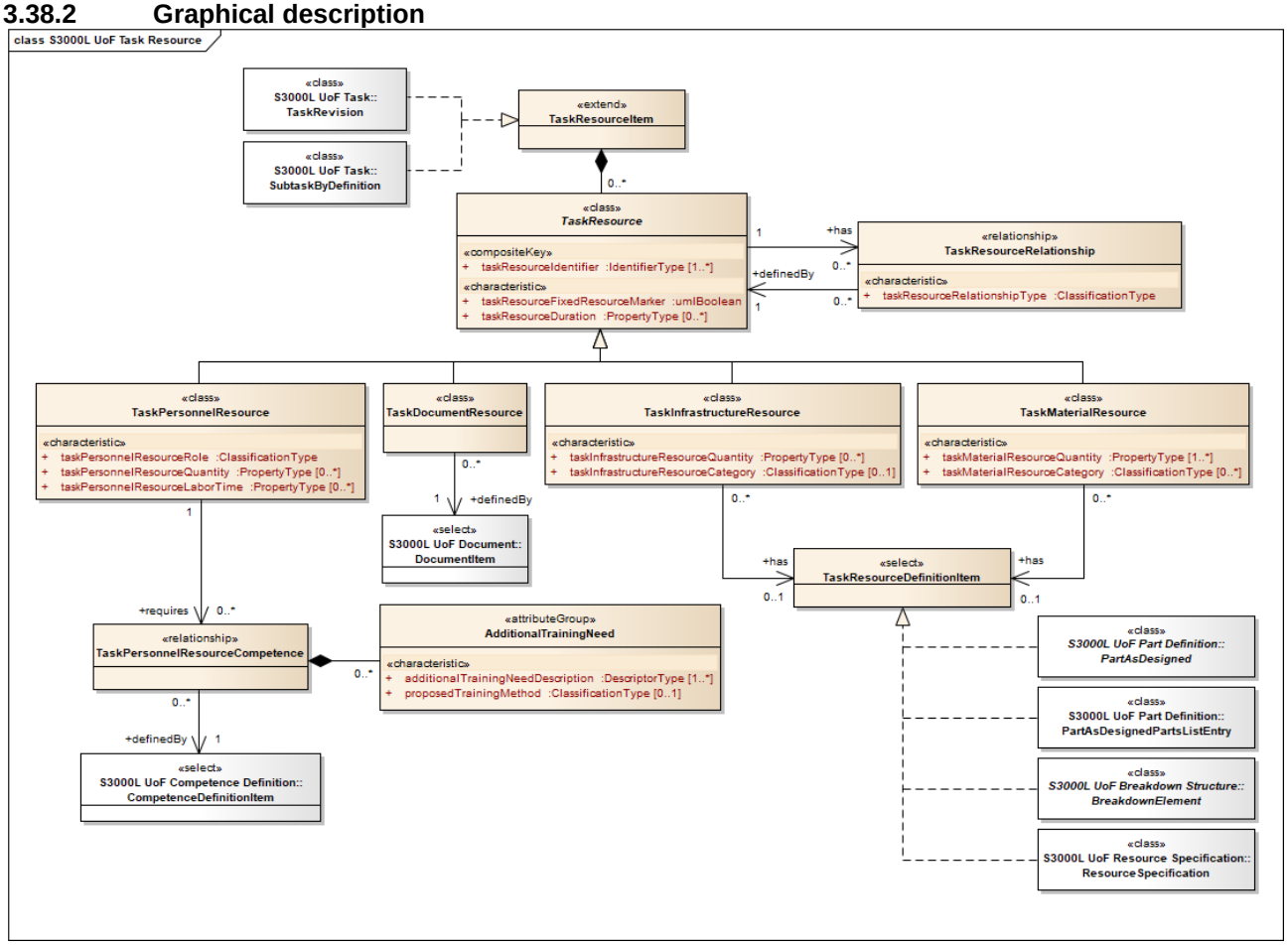
- ChangeControlledItem. Refer to Para 3.4
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.38 S3000L UoF Task Resource

3.38.1 Description

The Task Resource UoF supports a detailed specification of resources needed to perform a specified amount of work.

3.38.2 Graphical description



ICN-B6865-S3000L0228-004-01

Fig 42 S3000L UoF Task Resource

3.38.3 Class definition

3.38.3.1 AdditionalTrainingNeed

`AdditionalTrainingNeed` is an `<<attributeGroup>>` that specifies additional learning required for the associated `CompetenceDefinitionItem` (refer to Para 3.6) in order to qualify as the `TaskPersonnelResource`.

3.38.3.1.1 Attribute(s)

This `<<attributeGroup>>` has the following attributes:

- `additionalTrainingNeedDescription`, one or many
- `proposedTrainingMethod`, zero or one

3.38.3.2 TaskDocumentResource

`TaskDocumentResource` is a `TaskResource` that identifies a `Document` (refer to Para 3.11) used as a resource.

Example(s)

- A form that must be filled out before, during or after the specified amount of work is carried out.

3.38.3.2.1 Attribute(s)

This class has the following attributes:

- taskResourceIdentifier (inherited from TaskResource), one or many
- taskResourceFixedResourceMarker (inherited from TaskResource)
- taskResourceDuration (inherited from TaskResource), zero, one or many

3.38.3.2.2 Associations

This class has the following associations:

- A directed has association with zero, one or many instances of TaskResourceRelationship (inherited from TaskResource)
- A directed definedBy association with one instance of DocumentItem. Refer to Para 3.11

3.38.3.2.3 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem (inherited from TaskResource). Refer to Para 3.2
- ChangeControlledItem (inherited from TaskResource). Refer to Para 3.4
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Para 3.31

3.38.3.3 TaskInfrastructureResource

TaskInfrastructureResource is a TaskResource that defines foundational systems and services required as a resource.

3.38.3.3.1 Attribute(s)

This class has the following attributes:

- taskResourceIdentifier (inherited from TaskResource), one or many
- taskResourceFixedResourceMarker (inherited from TaskResource)
- taskResourceDuration (inherited from TaskResource), zero, one or many
- taskInfrastructureResourceQuantity, zero, one or many
- taskInfrastructureResourceCategory, zero or one

3.38.3.3.2 Associations

This class has the following associations:

- A directed has association with zero, one or many instances of TaskResourceRelationship (inherited from TaskResource)
- A directed has association with zero or one instance of TaskResourceDefinitionItem

3.38.3.3.3 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem (inherited from TaskResource). Refer to Para 3.2
- ChangeControlledItem (inherited from TaskResource). Refer to Para 3.4
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24

- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.38.3.4 TaskMaterialResource

TaskMaterialResource is a TaskResource that identifies parts which are required as a resource.

3.38.3.4.1 Attribute(s)

This class has the following attributes:

- taskResourceIdentifier (inherited from TaskResource), one or many
- taskResourceFixedResourceMarker (inherited from TaskResource)
- taskResourceDuration (inherited from TaskResource), zero, one or many
- taskMaterialResourceQuantity, one or many
- taskMaterialResourceCategory, zero, one or many

3.38.3.4.2 Associations

This class has the following associations:

- A directed has association with zero, one or many instances of TaskResourceRelationship (inherited from TaskResource)
- A directed has association with zero or one instance of TaskResourceDefinitionItem

3.38.3.4.3 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem (inherited from TaskResource). Refer to Para 3.2
- ChangeControlledItem (inherited from TaskResource). Refer to Para 3.4
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.38.3.5 TaskPersonnelResource

TaskPersonnelResource is a TaskResource that specifies the man power required as a resource.

3.38.3.5.1 Attribute(s)

This class has the following attributes:

- taskResourceIdentifier (inherited from TaskResource), one or many
- taskResourceFixedResourceMarker (inherited from TaskResource)
- taskResourceDuration (inherited from TaskResource), zero, one or many
- taskPersonnelResourceRole
- taskPersonnelResourceQuantity, zero, one or many
- taskPersonnelResourceLaborTime, zero, one or many

3.38.3.5.2 Associations

This class has the following associations:

- A directed has association with zero, one or many instances of TaskResourceRelationship (inherited from TaskResource)
- A directed requires association with zero, one or many instances of TaskPersonnelResourceCompetence.

3.38.3.5.3 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem (inherited from TaskResource). Refer to Para 3.2
- ChangeControlledItem (inherited from TaskResource). Refer to Para 3.4
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.38.3.6 TaskPersonnelResourceCompetence

TaskPersonnelResourceCompetence is a <<relationship>> that identifies the proficiency required for TaskPersonnelResource.

3.38.3.6.1 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of AdditionalTrainingNeed
- A directed definedBy association with one instance of CompetencyDefinitionItem. Refer to Para 3.6

3.38.3.6.2 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.38.3.7 TaskResource

TaskResource is a <<class>> that identifies means that have to be available to perform a specified amount of work.

3.38.3.7.1 Attribute(s)

This class has the following attributes:

- taskResourceIdentifier, one or many
- taskResourceFixedResourceMarker
- taskResourceDuration, zero, one or many

3.38.3.7.2 Associations

This class has the following associations:

- A directed has association with zero, one or many instances of TaskResourceRelationship

3.38.3.7.3 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- ChangeControlledItem. Refer to Para 3.4
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.38.3.8 TaskResourceDefinitionItem

TaskResourceDefinitionItem is a <<select>> interface that identifies items which can be used as either infrastructure or material resources.

3.38.3.8.1 Class members

This <<select>> interface includes the following class members:

- BreakdownElement. Refer to Para 3.3
- PartAsDesigned. Refer to Para 3.25
- PartAsDesignedPartsListEntry. Refer to Para 3.25
- ResourceSpecification. Refer to Para 3.32

3.38.3.9 TaskResourceItem

TaskResourceItem is an <<extend>> interface that provides its associated data model to those classes that implement it.

3.38.3.9.1 Class members

Classes that implement the TaskResourceItem <<extend>> interface are:

- SubtaskByDefinition. Refer to Para 3.36
- TaskRevision. Refer to Para 3.36

3.38.3.9.2 Associations

The TaskResourceItem <<extend>> interface has the following associations:

- An aggregate association with zero, one or many instances of TaskResource

3.38.3.10 TaskResourceRelationship

TaskResourceRelationship is a <<relationship>> where one TaskResource relates to another TaskResource.

3.38.3.10.1 Attribute(s)

This class has the following attributes:

- taskResourceRelationshipType

3.38.3.10.2 Associations

This class has the following associations:

- A directed definedBy association with one instance of TaskResource

3.38.3.10.3 Implementations

This class implements the following <<extend>> interfaces:

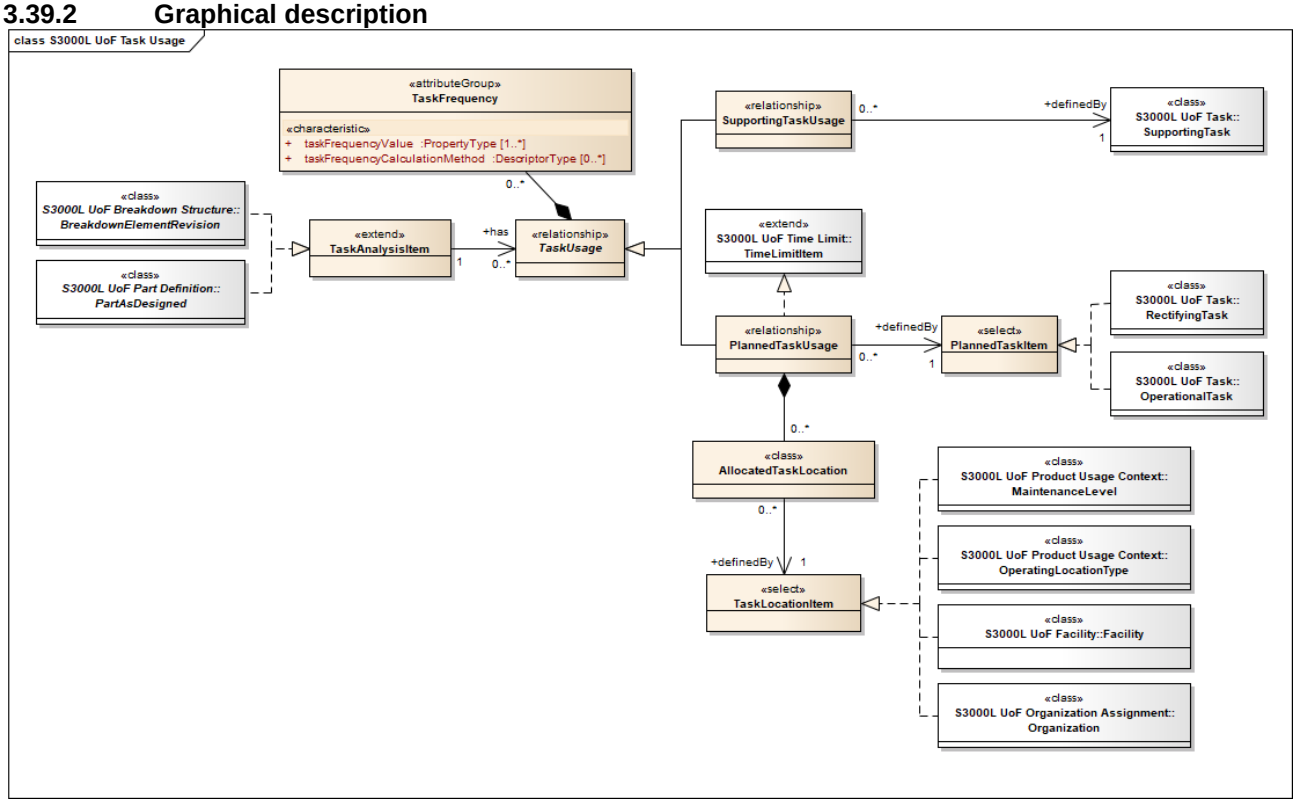
- ApplicabilityStatementItem. Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.39 S3000L UoF Task Usage

3.39.1 Description

The Task Usage UoF provides the capability to expand the definition for the execution of a task in the context of a given support solution.

3.39.2 Graphical description



ICN-B6865-S3000L0287-001-01

Fig 43 S3000L UoF Task Usage

3.39.3 Class definition

3.39.3.1 AllocatedTaskLocation

AllocatedTaskLocation is a <<class>> that identifies where a Task (refer to Para 3.36) is to be performed in the context of a given support solution.

3.39.3.1.1 Associations

This class has the following associations:

- A directed definedBy association with one instance of TaskLocationItem

3.39.3.1.2 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.39.3.2 PlannedTaskItem

PlannedTaskItem is a <<select>> interface that identifies items which can be used in the context of PlannedTaskUsage.

3.39.3.2.1 Class members

This <<select>> interface includes the following class members:

- OperationalTask. Refer to Para 3.36
- RectifyingTask. Refer to Para 3.36

3.39.3.3 PlannedTaskUsage

PlannedTaskUsage is a TaskUsage that expands the definition of a required Task (refer to Para 3.36) in the context of a given support solution.

Note

Both rectifying and operational tasks are justified by task requirements identified during various analysis activities.

3.39.3.3.1 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of TaskFrequency (inherited from TaskUsage)
- An aggregate association with zero, one or many instances of AllocatedTaskLocation
- A directed definedBy association with one instance of PlannedTaskItem

3.39.3.3.2 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem (inherited from TaskUsage). Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- TimeLimitItem. Refer to Para 3.40

3.39.3.4 SupportingTaskUsage

SupportingTaskUsage is a TaskUsage that expands the definition of an embedded reusable Task (refer to Para 3.36) in the context of a given support solution.

Note

A SupportingTask has no special characterizations apart from a TaskFrequency since it will never be performed on its own.

3.39.3.4.1 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of TaskFrequency (inherited from TaskUsage)
- A directed definedBy association with one instance of SupportingTask

3.39.3.4.2 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem (inherited from TaskUsage). Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.39.3.5 TaskAnalysisItem

TaskAnalysisItem is an <<extend>> interface that provides its associated data model to those classes that implement it.

3.39.3.5.1 Class members

Classes that implement the TaskAnalysisItem <<extend>> interface are:

- BreakdownElementRevision. Refer to Para 3.3
- PartAsDesigned. Refer to Para 3.25

3.39.3.5.2 Associations

The TaskAnalysisItem <<extend>> interface has the following associations:

- A directed has association with zero, one or many instances of TaskUsage

3.39.3.6 TaskFrequency

TaskFrequency is an <<attributeGroup>> that specifies the rate of occurrence of a Task (refer to Para 3.36) in its defined usage.

3.39.3.6.1 Attribute(s)

This <<attributeGroup>> has the following attributes:

- taskFrequencyValue, one or many
- taskFrequencyCalculationMethod, zero, one or many

3.39.3.6.2 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2

3.39.3.7 TaskLocationItem

TaskLocationItem is a <<select>> interface that identifies items where Tasks (refer to Para 3.36) can be performed.

3.39.3.7.1 Class members

This <<select>> interface includes the following class members:

- Facility. Refer to Para 3.13
- MaintenanceLevel. Refer to Para 3.29
- OperatingLocationType. Refer to Para 3.29
- Organization. Refer to Para 3.24

3.39.3.8 TaskUsage

TaskUsage is a <<relationship>> that identifies a Task (refer to Para 3.36) required for the TaskAnalysisItem.

3.39.3.8.1 Associations

This class has the following associations:

- An aggregate association with zero, one or many instances of TaskFrequency

3.39.3.8.2 Implementations

This class implements the following <<extend>> interfaces:

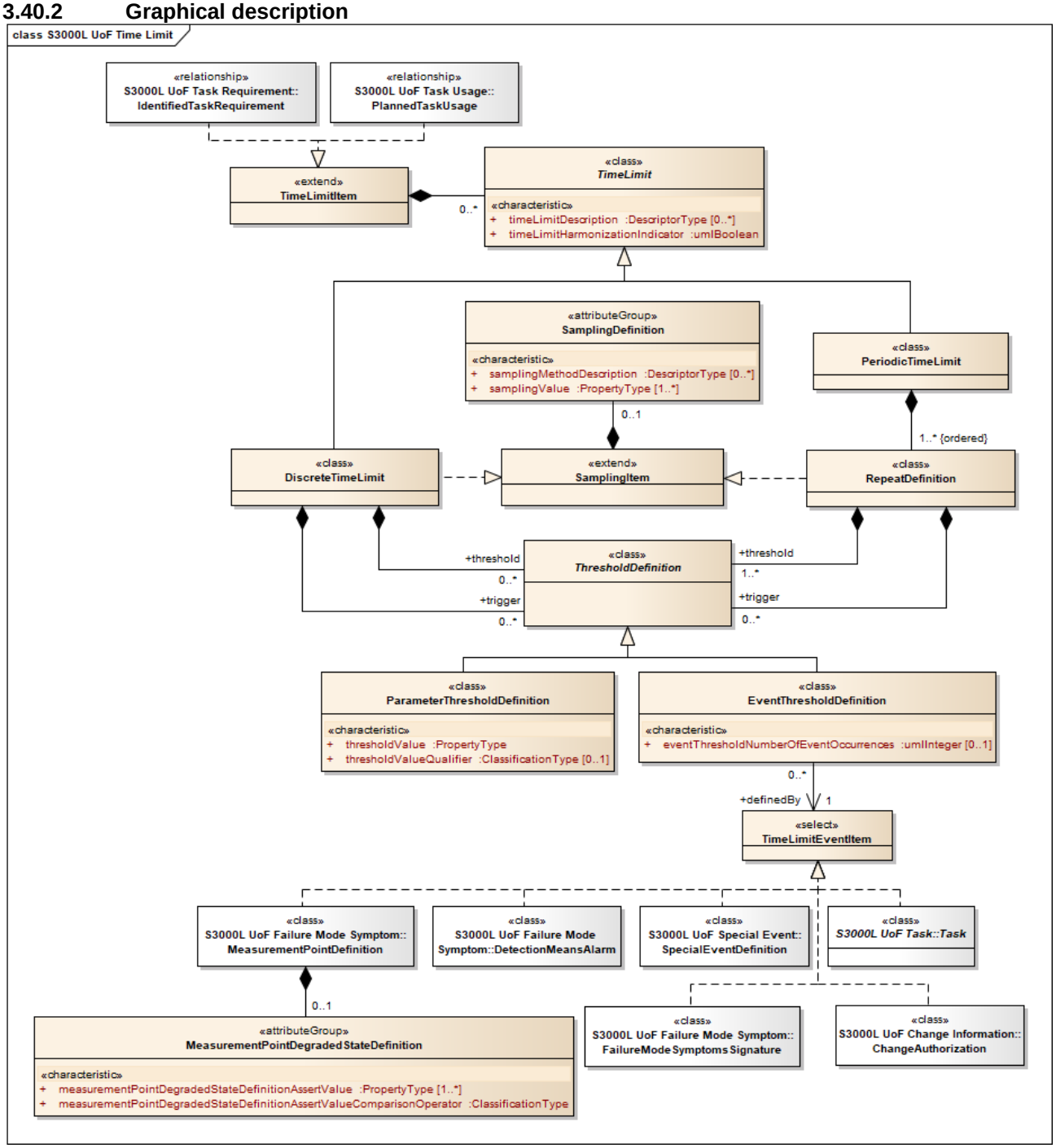
- ApplicabilityStatementItem. Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.40 S3000L UoF Time Limit

3.40.1 Description

The Time Limit UoF provides the capability to define the circumstances under which an action is to be initiated.

3.40.2 Graphical description



ICN-B6865-S3000L0288-001-01

Fig 44 S3000L UoF Time Limit

3.40.3 Class definition

3.40.3.1 DiscreteTimeLimit

`DiscreteTimeLimit` is a `TimeLimit` that is distinct, where its next possible occurrence cannot be scheduled.

Note

A Trigger is something that activates a `DiscreteTimeLimit`.

Note

A Threshold is a point that must not be exceeded once the DiscreteTimeLimit has been activated. If there is no threshold value, then the TimeLimitItem must be initiated immediately after the DiscreteTimeLimit activation.

3.40.3.1.1 Attribute(s)

This class has the following attributes:

- timeLimitDescription (inherited from TimeLimit), zero, one or many
- timeLimitHarmonizationIndicator (inherited from TimeLimit)

3.40.3.1.2 Associations

This class has the following associations:

- An aggregate threshold association with zero, one or many instances of ThresholdDefinition
- An aggregate trigger association with zero, one or many instances of ThresholdDefinition

3.40.3.1.3 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem (inherited from TimeLimit). Refer to Para 3.2
- ChangeControlledItem (inherited from TimeLimit). Refer to Para 3.4
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SamplingItem. Refer to Para 3.40

3.40.3.2 EventThresholdDefinition

EventThresholdDefinition is a ThresholdDefinition that is driven by occurrences of related TimeLimitEventItems.

3.40.3.2.1 Attribute(s)

This class has the following attributes:

- eventThresholdNumberOfEventOccurrences, zero or one

3.40.3.2.2 Associations

This class has the following associations:

- A directed definedBy association with one instance of TimeLimitEventItem

3.40.3.2.3 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem (inherited from ThresholdDefinition). Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D

- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.40.3.3 MeasurementPointDegradedStateDefinition

MeasurementPointDegradedStateDefinition is an <<attributeGroup>> that specifies a measurable condition which, when met, identifies that a degraded state is entered.

3.40.3.3.1 Attribute(s)

This <<attributeGroup>> has the following attributes:

- measurementPointDegradedStateDefinitionAssertValue, one or many
- measurementPointDegradedStateDefinitionAssertValueComparisonOperator

3.40.3.4 ParameterThresholdDefinition

ParameterThresholdDefinition is a ThresholdDefinition that is continuously measured and evaluated, and when reached, activates the associated trigger threshold.

3.40.3.4.1 Attribute(s)

This class has the following attributes:

- thresholdValue
- thresholdValueQualifier, zero or one

3.40.3.4.2 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem (inherited from ThresholdDefinition). Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.40.3.5 PeriodicTimeLimit

PeriodicTimeLimit is a TimeLimit that is repeated and its next occurrence can be scheduled.

3.40.3.5.1 Attribute(s)

This class has the following attributes:

- timeLimitDescription (inherited from TimeLimit), zero, one or many
- timeLimitHarmonizationIndicator (inherited from TimeLimit)

3.40.3.5.2 Associations

This class has the following associations:

- An ordered aggregate association with one or many instances of RepeatDefinition

3.40.3.5.3 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem (inherited from TimeLimit). Refer to Para 3.2
- ChangeControlledItem (inherited from TimeLimit). Refer to Para 3.4
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.40.3.6 RepeatDefinition

RepeatDefinition is a <<class>> that represents the set of circumstances which defines the PeriodicTimeLimit.

Note

A trigger is something that activates a RepeatDefinition.

Note

A threshold is a point that must not be exceeded by the TimeLimitItem once the RepeatDefinition has been activated.

Note

Only one RepeatDefinition must be active at a specific moment in time.

3.40.3.6.1 Associations

This class has the following associations:

- An aggregate threshold association with one or many instances of ThresholdDefinition
- An aggregate trigger association with zero, one or many instances of ThresholdDefinition

3.40.3.6.2 Implementations

This class implements the following <<extend>> interfaces:

- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SamplingItem

3.40.3.7 SamplingDefinition

SamplingDefinition is an <<attributeGroup>> that specifies whether the associated action is to be performed on a subset of the population of the associated target item.

3.40.3.7.1 Attribute(s)

This <<attributeGroup>> has the following attributes:

- samplingMethodDescription, zero, one or many
- samplingValue, one or many

3.40.3.8 SamplingItem

SamplingItem is an <<extend>> interface that represents the common behavior for those classes which can have an associated SamplingDefinition.

3.40.3.8.1 Class members

Classes that implement the SamplingItem <<extend>> interface are:

- DiscreteTimeLimit
- RepeatDefinition

3.40.3.8.2 Associations

The SamplingItem <<extend>> interface has the following associations:

- An aggregate association with zero or one instance of SamplingDefinition

3.40.3.9 ThresholdDefinition

ThresholdDefinition is a <<class>> that represents the circumstance that is used as a trigger or threshold.

3.40.3.9.1 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.40.3.10 TimeLimit

TimeLimit is a <<class>> that represents the specification of circumstances under which the associated item is initiated.

Note

Time is used in the sense of "time to do something" and must not be seen as only a period of time.

Note

TimeLimit does not have the concept of an identification which means that if there is a change to a TimeLimit for a TimeLimitItem, then all TimeLimits must be resent as part of a Message.

3.40.3.10.1 Attribute(s)

This class has the following attributes:

- timeLimitDescription, zero, one or many
- timeLimitHarmonizationIndicator

3.40.3.10.2 Implementations

This class implements the following <<extend>> interfaces:

- ApplicabilityStatementItem. Refer to Para 3.2
- ChangeControlledItem. Refer to Para 3.4
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31

3.40.3.11 TimeLimitEventItem

TimeLimitEventItem is a <<select>> interface that identifies which items can be used to define an EventThresholdDefinition.

3.40.3.11.1 Class members

This <<select>> interface includes the following class members:

- ChangeAuthorization. Refer to Para 3.4
- DetectionMeansAlarm. Refer to Para 3.16
- FailureModeSymptomsSignature. Refer to Para 3.16
- MeasurementPointDefinition. Refer to Para 3.16
- SpecialEventDefinition. Refer to Para 3.35
- Task. Refer to Para 3.36

3.40.3.12 TimeLimitItem

TimeLimitItem is an <<extend>> interface that provides its associated data model to those classes that implement it.

3.40.3.12.1 Class members

Classes that implement the TimeLimitItem <<extend>> interface are:

- IdentifiedTaskRequirement. Refer to Para 3.37
- PlannedTaskUsage. Refer to Para 3.39

3.40.3.12.2 Associations

The TimeLimitItem <<extend>> interface has the following associations:

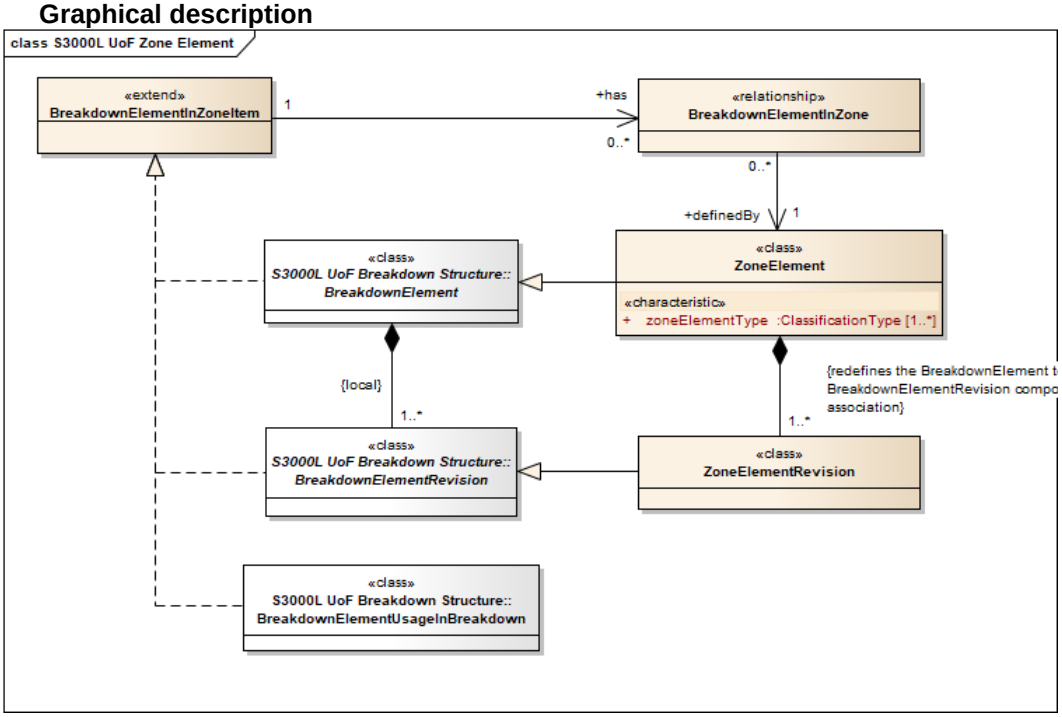
- An aggregate association with zero, one or many instances of TimeLimit

3.41 S3000L UoF Zone Element

3.41.1 Description

The Zone Element UoF defines the characteristics that are unique for a breakdown element that represents a three-dimensional space related to a Product.

3.41.2 Graphical description



ICN-B6865-S3000L0220-004-01

Fig 45 S3000L UoF Zone Element

3.41.3 Class definition

3.41.3.1 BreakdownElementInZone

`BreakdownElementInZone` is a `<<relationship>>` where a `BreakdownElementInZoneItem` relates to the `ZoneElement` where it is located.

3.41.3.1.1 Associations

This class has the following associations:

- A directed `definedBy` association with one instance of `ZoneElement`

3.41.3.1.2 Implementations

This class implements the following `<<extend>>` interfaces:

- `ApplicabilityStatementItem`. Refer to Para 3.2
- `ChangeControlledItem`. Refer to Para 3.4
- `DocumentReferencingItem` (inherited from `BaseObject`). Refer to Para 3.11
- `OrganizationReferencingItem` (inherited from `BaseObject`). Refer to Para 3.24
- `ProjectSpecificExtensionItem` (inherited from `BaseObject`). Refer to SX002D
- `RemarkItem` (inherited from `BaseObject`). Refer to Para 3.31

3.41.3.2 BreakdownElementInZoneItem

`BreakdownElementInZoneItem` is an `<<extend>>` interface that provides its associated data model to those classes that implement it.

3.41.3.2.1 Class members

Classes that implement the `BreakdownElementInZoneItem` `<<extend>>` interface are:

- BreakdownElement. Refer to Para 3.3
- BreakdownElementRevision. Refer to Para 3.3
- BreakdownElementUsageInBreakdown. Refer to Para 3.3

3.41.3.2.2 Associations

The BreakdownElementInZoneItem <<extend>> interface has the following associations:

- A directed has association with one instance of BreakdownElementInZone

3.41.3.3 ZoneElement

ZoneElement is a BreakdownElement (refer to Para 3.3) that represents a three-dimensional space related to a Product (refer to Para 3.27).

Note

A zone can also represent a work area such as a mechanical workshop onboard a ship.

3.41.3.3.1 Attribute(s)

This class has the following attributes:

- breakdownElementIdentifier (inherited from BreakdownElement), one or many
- breakdownElementName (inherited from BreakdownElement), zero, one or many
- breakdownElementEssentiality (inherited from BreakdownElement), zero or one
- zoneElementType, one or many

3.41.3.3.2 Associations

This class has the following associations:

- An aggregate association with one or many instances of ZoneElement

3.41.3.3.3 Implementations

This class implements the following <<extend>> interfaces:

- AnalysisCandidateItem (inherited from BreakdownElement). Refer to Para 3.21
- BreakdownElementInZoneItem (inherited from BreakdownElement
- DecisionTreeAnalysisItem (inherited from BreakdownElement). Refer to Para 3.8
- DetectionMeansAlarmItem (inherited from BreakdownElement). Refer to Para 3.16
- DigitalFileReferencingItem (inherited from BreakdownElement). Refer Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- FailureModeAnalysisItem (inherited from BreakdownElement). Refer to Para 3.14
- InServiceOptimizationAnalysisItem (inherited from BreakdownElement). Refer to Para 3.18
- MeasurementPointDefinitionItem (inherited from BreakdownElement). Refer to Para 3.16
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24

- PerformanceParameterItem (inherited from BreakdownElement). Refer to Para 3.26
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- SecurityClassificationItem (inherited from BreakdownElement). Refer to Para 3.33

3.41.3.4 ZoneElementRevision

ZoneElementRevision is a BreakdownElementRevision (refer to Para 3.3) representing an iteration applied to a ZoneElement.

3.41.3.4.1 Attribute(s)

This class has the following attributes:

- breakdownElementRevisionIdentifier (inherited from BreakdownElementRevision)
- breakdownElementDescription (inherited from BreakdownElementRevision), zero, one or many
- maintenanceSignificantOrRelevant (inherited from BreakdownElementRevision)
- breakdownElementRevisionRationale (inherited from BreakdownElementRevision), zero, one or many
- breakdownElementRevisionDate (inherited from BreakdownElementRevision), zero or one
- breakdownElementRevisionStatus (inherited from BreakdownElementRevision), zero or one

3.41.3.4.2 Associations

This class has the following associations:

- A directed has association, to zero, one or many objects of type BreakdownElementRevisionRelationship (inherited from BreakdownElementRevision)

3.41.3.4.3 Implementations

This class implements the following <<extend>> interfaces:

- AnalysisCandidateItem (inherited from BreakdownElementRevision). Refer to Para 3.21
- BreakdownElementInZoneItem (inherited from BreakdownElementRevision). Refer to Para 3.41
- ChangeControlledItem (inherited from BreakdownElementRevision). Refer to Para 3.4
- DetectionMeansAlarmItem (inherited from BreakdownElementRevision). Refer to Para 3.16
- DigitalFileReferencingItem (inherited from BreakdownElementRevision). Refer to Para 3.10
- DocumentReferencingItem (inherited from BaseObject). Refer to Para 3.11
- FailureModeAnalysisItem (inherited from BreakdownElementRevision). Refer to Para 3.14

- InServiceOptimizationAnalysisItem (inherited from BreakdownElementRevision). Refer to Para 3.18
- MeasurementPointDefinitionItem (inherited from BreakdownElementRevision). Refer to Para 3.16
- OrganizationReferencingItem (inherited from BaseObject). Refer to Para 3.24
- PerformanceParameterItem (inherited from BreakdownElementRevision). Refer to Para 3.26
- ProjectSpecificExtensionItem (inherited from BaseObject). Refer to SX002D
- RemarkItem (inherited from BaseObject). Refer to Para 3.31
- TaskAnalysisItem (inherited from BreakdownElementRevision). Refer to Para 3.39
- TaskRequirementAnalysisItem (inherited from BreakdownElementRevision). Refer to Para 3.37

Chapter 20

Data exchange

Table of contents

List of tables

List of figures

References

Table 1 References

Chap No./Document No.	Title
Chap 19	Data model
SX005G	S-Series ILS specification XML schema implementation guidance
http://www.plcs.org/plcslib/plcslib/	PLCS Platform Specific Model (PSM)
http://www.s3000l.org	S3000L website
ISO 10303-239 PLCS	Product Life Cycle Support (PLCS)

1 General

This chapter defines the standard technology used for the exchange of data that results from Logistics Support Analysis (LSA) as defined in S3000L. The exchange of data for S3000L is defined using XML and XML schema.

It is also possible to use the XML schema to receive data from sources, which provide input to the S3000L process.

S3000L XML schemas are published separately on the S3000L website (www.s3000l.org).

1.1 Purpose

The purpose of this chapter is to describe how the S3000L XML schemas support the S3000L LSA process and its interaction with other business processes.

1.2 Scope

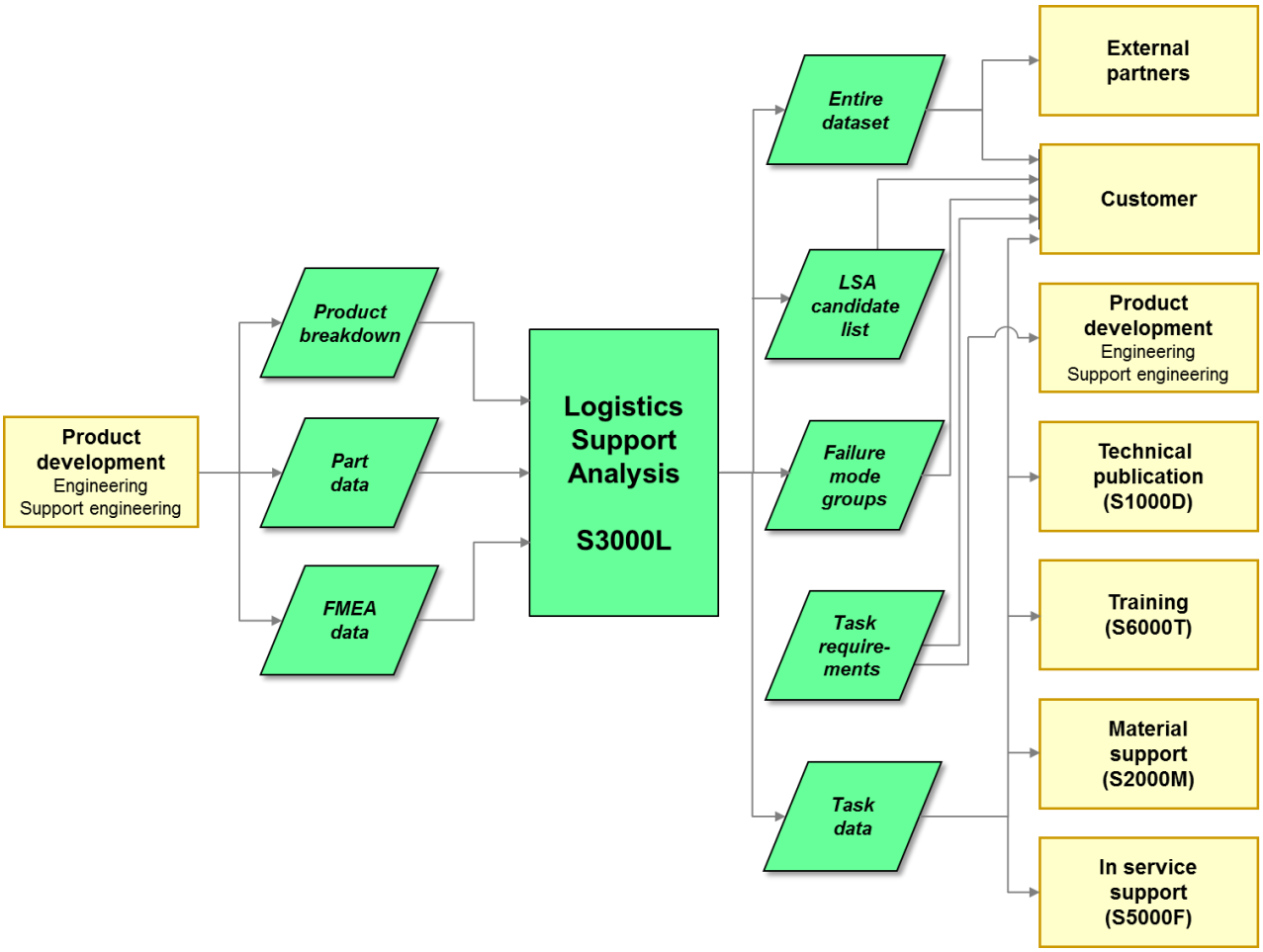
The following areas are within scope for this chapter:

- overview of S3000L data exchange using S3000L XML schemas
- overview of the defined S3000L XML schemas
- relationship between S3000L XML schemas and ISO 10303-239 PLCS

2 Overview

2.1 S3000L data exchange

Fig 1 provides an example of the types of data exchange supported by the S3000L XML schemas:



ICN-B6865-S3000L0236-003-00

Fig 1 Data exchange using S3000L XML schemas

2.1.1 Data flow from other disciplines to LSA

From product design and development to LSA, examples on types of data exchange using the S3000L XML schema, which can feed the S3000L activities, can contain the following information:

- Product breakdown can include, but is not limited to the following information on:
  - the Product that requires support and its Product variants
  - breakdowns defined for the Product, including its breakdown elements (eg, systems, functions, hardware, software, zones)
  - possible breakdown element realizations in terms of hardware or software
  - allowed Product design configurations
  - key performance parameters defined for the Product and its included elements
- part data can include, but are not limited to, the following information on:
  - part identifiers (eg, part numbers)
  - part names
  - technical characteristics
  - parts lists, Bill of Material (BOM)
  - operational authorized life
- Failure Modes and Effects Analysis (FMEA) data can include identified failure modes defined for parts that can require the identification of corrective maintenance tasks

2.1.2 Data flow from LSA to other disciplines or stakeholders

The identified types of data exchange (refer to Fig 1) contain the following information:

- the entire dataset (from LSA to external partners) contains the complete LSA dataset
- LSA candidate list (from LSA to the customer) includes information on selected LSA candidates and recommended (agreed upon) analysis activities
- failure mode groups (from LSA to the customer) include information about how to justify the recommended corrective maintenance tasks
- task requirements (from LSA to customers and product development, ie engineering and support engineering), includes, but is not limited to, information on:
  - task requirement specification
  - task requirement authority
  - task limit requirements
- task data (from LSA to the customer, technical publication, training, material support and in-service data feedback) include, but is not limited to, information on:
  - task identification
  - task justification
  - subtasks
  - task resources

3 S3000L XML schemas

The S3000L XML schemas are derived from the S3000L data model defined in Chap 19. The mapping of the S3000L data model to the S3000L XML schemas complies with the XML schema authoring rules defined by the Data Model and Exchange Working Group (DMEWG).

The option to only exchange updates is one of the defined features for the exchange of LSA data using the XML schemas for S3000L. Apart from complete (baseline) messages, it is possible to send update messages, which can accommodate minor, as well as major, changes to the LSA data. This means that the receiver of LSA data does not need to analyze what actions to take with respect to updating the target data set (eg, database). For more information, refer to SX005G.

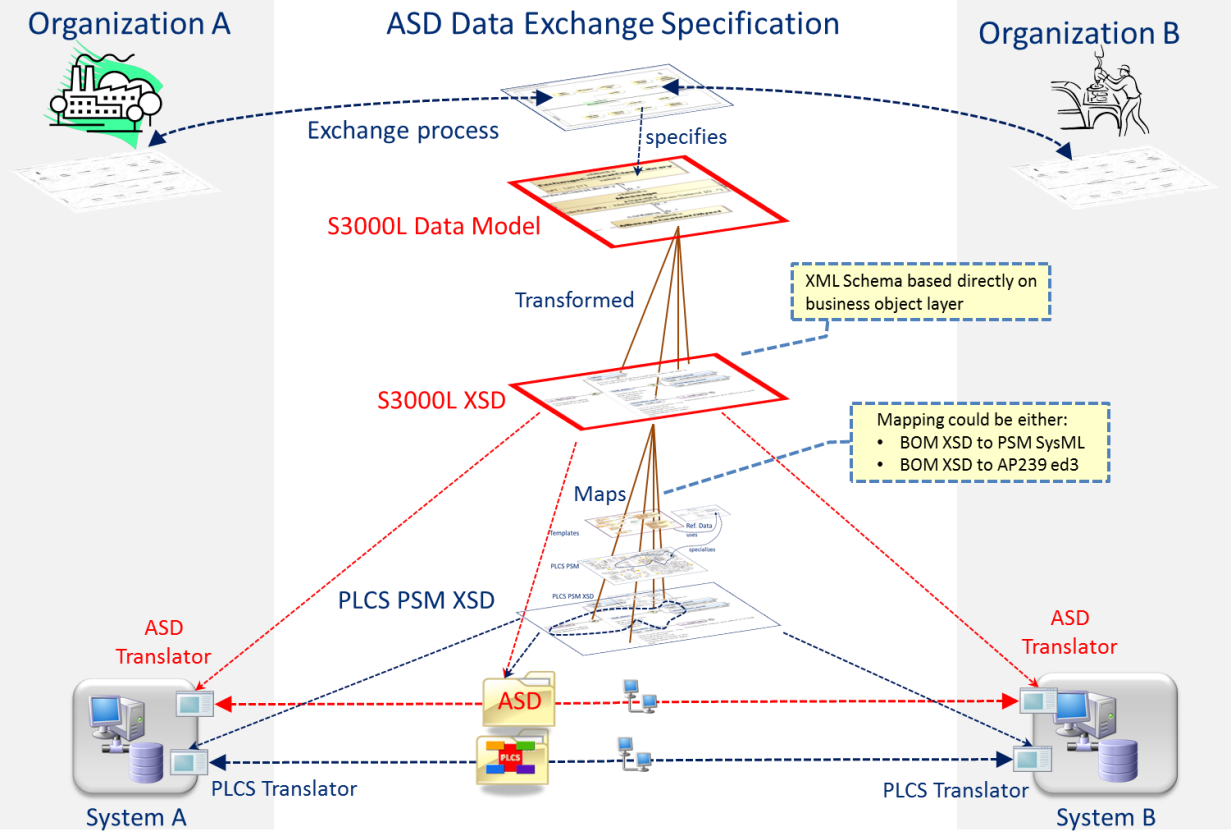
Another additional feature in the XML schemas for S3000L is the support of project-specific attributes, as described in Chap 19. This allows for projects to add data elements required for

the project, but which are not part of the S3000L data model and its associated XML schemas. For more information, refer to SX005G.

4 Product Life Cycle Support (PLCS)

S3000L XML schemas will provide mappings to ISO 10303-239 PLCS (ed3) in order to support continued use of ISO 10303-239 PLCS. Because of their design, these mappings will form an integral part of the respective S3000L XML schemas.

In awaiting ISO 10303-239 PLCS (ed3) and its associated data exchange development environment, all S-Series IPS specifications follow the same XML schema approach as defined for S3000L. Fig 2 illustrates how to consider an S-Series IPS specification XML schema with respect to ISO 10303-239 PLCS and OASIS PLCS Platform Specific Model (PSM).



ICN-B6865-S3000L0237-002-00

Fig 2 ASD XML schema to PLCS implementation mapping

The S-Series IPS specifications XML schemas aim to support data exchange at the Business Object Model (BOM) layer. Future XML schema will also include the mapping details required for an unambiguous mapping to PLCS of each element and attribute, in order to enable PLCS-based data exchange and/or PLCS-based data consolidation.

Chapter 21

Terms, abbreviations and acronyms

Table of contents

List of tables

References

Table 1 References

Chap No./Document No.	Title
None	

1 General

1.1 Purpose

This chapter provides a comprehensive dictionary for the terms, abbreviations and acronyms used throughout S3000L.

1.2 Scope

This chapter comprises a comprehensive dictionary of the terms, abbreviations, and acronyms (including initialisms and portmanteaus) used throughout this specification.

2 Glossary of terms

Table 2 provides the definitions of terms used throughout this specification.

Table 2 Glossary of terms

Term	Definition
allowed Product configuration	An approved combination of hardware and software parts that can or must be installed in specific locations or positions, and a set of associated engineering instructions to follow during assembly and operation. The allowed Product configuration demonstrates that a Product complies with applicable regulations.
alternate part	A part that can replace another part for any purpose, is context independent, and is equivalent to the original part in form, fit and function. – Refer to form, fit and function
availability	When the mission or operation is called for at an unknown time, availability measures the extent to which an item is in an operable and ready-for-use state at the start of a mission or operation.
Bill of Material (BOM)	A comprehensive list of the raw materials, assemblies, subassemblies, parts, and components needed to manufacture a Product, along with the quantities of each.
breakdown	A view of a Product that identifies the specific partitioning of the Product into a set of related elements, forming parent-child views of Product elements.
Breakdown Element (BE)	A definition of an item used to represent a system, subsystem, function, zone, hardware item, or any other type of Product partitioning. A breakdown element may be a member of one or more breakdowns.
Breakdown Element Identifier (BEI)	Identifies an individual breakdown element defined within a functional, physical, or any other type of Product breakdown. In the case of a physical breakdown element, there is also an indication of the installation location.
breakdown element realization	The combination of a hardware or software item with a breakdown element and its associated installation location. – Refer to realization
Breakdown Element Revision (BER)	Defines a specific release of a breakdown element. – Refer to revision
Breakdown Revision (BR)	Defines a specific release of a breakdown. – Refer to revision

Term	Definition
Built-In Test (BIT)	An integrated testing method implemented in an item. Usually, three types of BIT are implemented: – Power-on BIT (P-BIT), which is executed when the item starts up – Continuous BIT (C-BIT), which is executed periodically during the operation of the item, without any intervention from the operating crew – Initiated BIT (I-BIT), which is executed on demand by the operator or maintainer Each of these tests detects specific categories of failures, characterized by: – A detection rate, which is the percentage of functions or items whose failure is detected – A localization rate, also known as an isolation rate, which is the percentage of detected failures that are associated with the failure of a replaceable unit, which is identified unambiguously – A false alarm rate, which is the percentage of positive tests that do not actually represent a failure The intent of a BIT is to detect failures and localize replaceable units that have failed. Localization can be either non-ambiguous or ambiguous. – A non-ambiguous failure means it is possible to attribute the failure to a specific item with an acceptable degree of certainty – An ambiguous failure can be attributed to a group of replaceable parts without certainty as to which specific part in the group has failed The goal of troubleshooting analysis is to remove the ambiguity behind common or critical failures. For critical failures, the BIT typically issues a warning or alarm message to the operator.
Built-in Test Equipment (BITE)	Test equipment permanently installed in the Product, for the express purpose of testing the Product. – Refer to General Purpose Test Equipment (GPTE), special test equipment, test equipment
Candidate Item (CI)	An item identified as requiring analysis through the LSA process.
Candidate Item List (CIL)	A list of the candidate items that serves as documentation of the LSA candidate selection process. – Refer to Candidate Item (CI)
Chemical Abstracts Service (CAS) identification number	A unique identifier of a substance, suitable for lookup in the CAS database. It is often referred to as the CAS number.

Term	Definition
Commercial Off-The-Shelf (COTS)	Software or hardware that is ready-made and available for sale, lease, or license to the public. This term is typically used in reference to technology products.
common cause	A failure cause which results in multiple related failures. For example, the failure of a power supply leads to the failure of all items to which it supplies power. When analyzing all of these failures together, the detected common cause is the failure of the power supply.
Concept of Operations (CONOPS)	A document describing the characteristics of a proposed Product from the perspective of a Product user. It is used to communicate the desired Product characteristics to all stakeholders, in both a quantitative and qualitative manner.
consumable	A non-repairable item, typically purchased and stocked in bulk, and is disposable (eg, fluids, grease, screws, O-rings).
corrective maintenance	A maintenance activity carried out to restore a malfunctioning item to its full functionality.
Customer Requirements Document (CRD)	A document containing all of the customer's supportability requirements. The CRD is a necessary input for the LSA Guidance Conference (LSA GC).
damage	A loss or reduction of functionality, not including failures due to the intrinsic reliabilities of an item. Normally, damage will require a maintenance task to restore the damaged item to its full functionality. Different types of damage (ie, ways in which an item can be damaged) are often grouped into families, which are addressed by a standard repair procedure. For example, a structure can have scratches, dents, and/or cracks, but it is possible to define a single standard repair procedure for any such structural damage.
Data and Assumptions Definition Document (DADD)	A document containing the ground rules and assumptions for Life Cycle Cost (LCC) analysis. This includes organizational, program, technical and economic data.
dangerous substance	A substance known to harm people and/or the environment.
Data Element List (DEL)	A list of selected data elements, generally given by the output of a data element tailoring process. This list contains all the data elements required for a project, and is not necessarily limited to data elements pre-defined by any particular information standard.
defect	Any nonconformity of an item to its specified requirements. A defect does not necessarily result in a failure of the item. – Refer to failure

Term	Definition
definition files	Support files related to an item, such as technical descriptions, interface definitions, wiring diagrams or protocols. Definition files can be useful when it becomes necessary to carry out functional or electrical tests to help isolate the failed item if the performance of simple troubleshooting fails to do so.
demilitarization	An operation that makes a defense Product unavailable or unsuitable for its intended military use.
deviation	The authorization to deviate from a particular requirement of an approved Product/equipment configuration for a specified period. Deviations allow the acceptance of a Product/equipment that does not fully meet a particular requirement, but is considered suitable for use either as-is, after repair, or after modification by an approved method.
Diminishing Manufacturing Sources (DMS)	The loss or impending loss of item manufacturers or suppliers. – Refer to Diminishing Manufacturing Sources and Material Shortages (DMSMS)
Diminishing Manufacturing Sources and Material Shortages (DMSMS)	The loss or impending loss of item manufacturers, suppliers, or raw materials. – Refer to Diminishing Manufacturing Sources (DMS)
directly replaceable	A hardware and software element categorization that indicates the item is directly replaceable on the Product. – Refer to not directly replaceable, non-replaceable
disposal	The act of ending the service life of a Product. – Refer to disposal phase, disposal process
disposal phase	The final phase of the Product life cycle, in which the disposal of the Product is achieved. – Refer to disposal process
disposal process	A set of tasks to be performed on the Product in order to achieve the disposal of the Product when it reaches the end of its service life. It is necessary to consider the disposal process from the very early phases of every project in which a Product is acquired, even though it is performed at the very end of the Product life cycle. – Refer to disposal, disposal phase
End Item Acronym Code (EIAC)	A code used to identify a Product in several logistics product data specifications, for example, refer to GEIA-STD-0007. – Refer to Product
external cause	A cause due to an event independent of normal Product usage, such as a bird strike for an aircraft. – Refer to internal cause, special event

Term	Definition
facility	A place suited for a particular purpose, such as repair, maintenance, training, or other supportability activities.
failure	A failure occurs when an item does not perform to the level required by its performance specification under normal use and prevents further use of the item.
failure cause	Any circumstance leading to a particular failure that occurred during Product design, manufacture or use.
failure detection	The method by which an operator becomes aware of a failure.
failure mode	The end result of a predicted or observed physical, mechanical, thermal or other process whose end result can or has led to a failure. The failure mode is stated relative to the operating conditions at the time of the failure.
failure rate	The probability of a failure occurring. It is expressed in failures per unit of operating time.
Failure Modes and Effects Analysis (FMEA)	A procedure for analyzing the potential failure modes of a system to classify them by severity, or to determine the effect of lower-level failures on the entire system.
Failure Mode, Effects, and Criticality Analysis (FMECA)	An extension of a FMEA that also includes a criticality analysis. The purpose of a criticality analysis is to chart the probability of failure modes against the severity of their consequences. It is constructed by calculating how probable it is for each particular fault in each particular item to have caused each failure mode. The severity of the consequences of each fault is used to determine the remedial action required to correct the fault and return the item to service.
fault	A physical condition that causes a device, a component, or an element to fail to perform in accordance with the item requirements.
fault isolation procedure	The process of determining the location of a fault to the extent necessary to repair the faulty item.
Field Loadable Software (FLS)	Software that can be installed into one or several pieces of equipment of a Product, without the need to remove the equipment from its installed location. – Refer to firmware, Shop Loadable Software (SLS)
firmware	Software that can be loaded into a Line Replaceable Unit or Shop Replaceable Unit, but requires the host component to be removed from its installation location on the operational system and subsequently replaced. – Refer to Field Loadable Software (FLS), Shop Loadable Software (SLS)

Term	Definition
form, fit and function	The physical parameters (form), physical interface (fit) and performance requirements (function) of an item. Usually, this phrase is used to express the suitability of a similar item that, if it were to replace the original item in a Product or piece of equipment, it would not alter the performance characteristics of that Product or piece of equipment.
functional analysis	The identification and characterization of the functions performed by the Product, systems, subsystems and sub-subsystems. It is possible to characterize each system function using the parameters that: – it takes as inputs – it requires as part of its operation – it provides as outputs The outputs provided by the characterization justify the inclusion of the items in the design. The results of the functionality analysis are used to identify the effect of a failed physical item or items on other associated functions. The results also indicate the detection methods and corrective actions required to prevent failure, and ensure the continuity or restoration of the item's intended functionality.
functional breakdown	A functional breakdown refers to the partitioning of functions typically provided by the Product from a user or operator point of view. For example, an aircraft's functional breakdown can include elements for taxiing, take off, flying, or landing. – Refer to system breakdown
functional symptom	– Refer to symptom
General Purpose Test Equipment (GPTE)	Test equipment that is not limited to a specific test application. – Refer to Built-In Test Equipment (BITE), special test equipment, test equipment
Guidance Conference (GC)	A conference where the contractor and the customer agree on the scope and depth of the LSA business processes and procedures.
Guidance Document (GD)	A signed, contractual record of the agreements reached by the contractor and the customer during the LSA GC.
hardware element	A BE representing a specification realized as a hardware part.
infrastructure	An overarching term for all resources required for an activity but not specified. For example, "facilities and infrastructure" includes the subject facilities, and any other improvements required to use the facilities, such as roads or utilities.

Term	Definition
Integrated Logistics Support (ILS)	An integrated and iterative process for developing material and a support strategy that optimizes functional support, leverages existing resources, and guides the system engineering process to quantify and lower life cycle cost and decrease the logistics footprint (ie, the demand for logistics), making the system easier to support. – Refer to Integrated Product Support (IPS) **Note** Integrated Logistics Support (ILS) has been replaced by Integrated Product Support (IPS).
Integrated Product Support (IPS)	An integrated and iterative process for developing material and a support strategy that optimizes functional support, leverages existing resources, and guides the system engineering process to quantify and lower life cycle cost and decrease the logistics footprint (ie, the demand for logistics), making the system easier to support. – Refer to Integrated Logistics Support (ILS)
internal cause	A cause due to an event that results from normal Product usage, (eg, excessive vibration). – Refer to external cause, special event
Key Performance Indicator (KPI)	A crucial Product requirement, expressed in a quantitative manner.
labor time	The accumulated time of personnel working on an entire task or an individual subtask. If the persons involved are of different competencies, the labor time is reported separately for each area of competency.
Level of Repair Analysis (LORA)	An analytical methodology performed on a list of selected LSA candidates to determine an optimized maintenance solution. It takes into consideration customers' support and operational requirements, Product technical information, economic aspects, and legal and environmental constraints. The outcome indicates at which maintenance level each selected item must be removed, replaced, repaired, checked, overhauled, inspected or discarded.
life cycle	The total lifespan of a Product, from its initial feasibility phase through its service, withdrawal, and disposal phases.
Life Cycle Cost (LCC)	All direct costs plus any indirect-variable costs associated with the procurement, operations, support and disposal of the Product. – Refer to Total Cost of Ownership (TCO), Whole Life Cost (WLC)
Line Replaceable Unit (LRU)	An item that can be removed and installed at the organizational maintenance, or field level. Synonymous with "Weapon Replacement Assembly (WRA)" and "module replaceable unit".
localization	Failure isolation that indicates the item or group of items that have failed. This isolation is usually associated with a failure detection.

Term	Definition
Logistics Support Analysis (LSA)	A structured approach to increasing maintenance efficiency and reducing the cost of providing support by planning all aspects of IPS as far in advance as possible.
LSA candidate item	– Refer to Candidate Item (CI)
LSA database	A database that records the supportability data gathered through the LSA process. Historically synonymous with “Logistics Support Analysis Record (LSAR)” and “Logistics Product Database”.
LSA Guidance Conference (LSA GC)	A conference where the contractor and the customer agree on the scope and depth of the LSA process and procedures.
maintainability	A measure of the ability of an item to be retained in, or restored to a specified condition, when maintenance is performed using prescribed procedures and resources at each prescribed level of maintenance and repair.
maintenance concept	A strategy for providing maintenance support to ensure that a Product meets its mission performance requirements.
maintenance plan	A plan for conducting maintenance in the most efficient and cost-effective way, such that the Product performs as intended to meet its mission requirements. The maintenance plan also ensures that all required maintenance resources are in place to support the deployment of the Product. The maintenance plan specifies the time, place and method for the performance of maintenance tasks on the Product/equipment, including both preventive and corrective maintenance.
Maintenance Task Analysis (MTA)	A detailed analysis process that results in the identification of the procedures, spares and materials, tools, support equipment, personnel skill levels, facilities, and estimated and elapsed times, that it is necessary to consider for each corrective, repair or preventive maintenance task.
Maintenance Relevant Item (MRI)	An item that can be repaired or replaced in case of a failure or damage. Normally, this item is automatically a potential LSA candidate, but not necessarily a Maintenance Significant Item (MSI). – Refer to LSA candidate, Maintenance Significant Item (MSI)
Maintenance Significant Item (MSI)	An item identified by any selection process to be the result of a Preventive Maintenance Analysis (PMA) procedure (eg, described in S4000P). An MSI can become an LSA candidate if the PMA process identifies Preventive Maintenance Task Requirements Interval (PMTRI) documented in the LSA database. Refer to S4000P.
Mean Elapsed Time (MET)	The average time required to complete a task, irrespective of the number of personnel involved. The duration of the whole task is the sum of the time required to perform each of the subtasks and the referenced supporting tasks.

Term	Definition
Mean Time Between Failures (MTBF)	MTBF is the predicted elapsed time between inherent failures of a system during operation. It is calculated as the arithmetic mean (average) time between failures of a system. The MTBF is typically part of a model that assumes the failed system is immediately repaired (zero elapsed time) as a part of a renewal process. In contrast, the Mean Time To Failure (MTTF) measures the average time between failures with the modeling assumption that the failed system is not repaired.
non-replaceable	A hardware and software element categorization that indicates an item is intrinsically related to another item and cannot be replaced separately. – Refer to directly replaceable, not directly replaceable
not directly replaceable	A hardware and software element categorization that indicates an item is not directly replaceable on the Product without the removal of the parent assembly. – Refer to directly replaceable, non-replaceable
Operational Requirements Document (ORD)	Defines the operational requirements in a qualitative way for the Product, and must be available for the LSA GC.
operational support task	Any support activity that resolves a task requirement in the context of the Product operation, but that does not relate to the areas of the direct operation or maintenance of the Product. For example, Packaging, Handling, Storage and Transportation (PHST) tasks are operational support tasks.
parts list	– Refer to Bill of Material (BOM)
personnel	People with specific human aptitudes, knowledge, skills, abilities and experience levels needed to perform Product operation and support tasks properly.
physical breakdown	A breakdown of the Product hardware and software based on the engineering design model/drawings. – Refer to breakdown
physical symptom	– Refer to symptom
preventive maintenance	Maintenance activities for preventing critical failures or damage in conjunction with safety, economical or ecological considerations. The term “preventive maintenance” also includes activities performed after special events, even though the occurrence of these events, chronological intervals between them, and other typical thresholds cannot be defined.
Preventive Maintenance Analysis (PMA)	The method used to determine the Preventive Maintenance Task Requirements (PMTR) and their intervals/thresholds.
Product	Any platform, system or equipment (air, sea, land vehicle, equipment or facilities, civil or military).

Term	Definition
product breakdown	– Refer to breakdown
product variant	A member of a Product family that is configured for a specific purpose and is offered to customers.
project	The task to develop, maintain and dispose of the Product.
realization	The completion of a physical breakdown from a design breakdown, including different physical systems, subsystems or components that can be used for the same installed location.
rectifying task	A rectifying task is any support activity that resolves an issue, such as failures, damages, special events, or thresholds.
reliability	The duration or probability of failure-free performance of a Product under stated conditions, or the probability that an item can perform its intended function for a specified period under stated conditions.
Reliability Centered Maintenance (RCM)	A disciplined logic or methodology used to identify scheduled maintenance tasks in order to maintain the inherent reliability of equipment, with a minimum expenditure of resources.
revision	A particular version of something that is undergoing an iterative process. The BR and BER capture the design iterations of a Product, the compilation of which represents a record of the Product's development progress. The use of these revisions supports the Configuration Management (CM) discipline, particularly with regard to the synchronization of changes originating from the Product design process, and their effect on the LSA process. – Refer to Breakdown Revision (BR), Breakdown Element Revision (BER)
scheduled maintenance	Maintenance activities that prevent the occurrence of critical failures or damages in conjunction with safety, economical or ecological aspects. These maintenance tasks have a corresponding interval or threshold (eg, after a certain amount of elapsed time, cycles completed, rounds expended, or distance traveled). Scheduled maintenance is a subset of preventive maintenance.
Shop Loadable Software (SLS)	Software that can be installed on an LRU, but which requires the target LRU to be removed from the system where it is installed. – Refer to Field Loadable Software (FLS), firmware
Shop Replaceable Unit	An item that requires a maintenance level higher than the organizational level for replacement. – Refer to Line Replaceable Unit (LRU)
Software Support Analysis (SSA)	The method used to identify a cost-effective Software Support Concept (SSC). – Refer to Software Support Concept (SSC)

Term	Definition
Software Support Concept (SSC)	A strategy for providing operational and maintenance support to software items. This includes the identification of all equipment, tools, personnel, documentation, infrastructure, skills, training and modification required to enable the continued usage of the software within the Product.
spare part	A part not installed on the Product and reserved for installation on the Product during the performance of future maintenance tasks.
special event	An event that occurs during a Product's life that is not part of the normal operation of the Product. A special event is generally due to either an external cause or the use of the Product outside its prescribed operating boundaries (such as the result of an over-G maneuver of an aircraft). – Refer to external cause, internal cause
special test equipment	Test equipment designed for a specific application. – Refer to Built-in Test Equipment (BITE), General Purpose Test Equipment (GPTE), test equipment
Structural Detail (SD)	A localized area of a Structure Significant Item (SSI). Refer to S4000P. – Refer to Structure Significant Item (SSI)
Structural Item (SI)	Any part of the Product structure.
Structure Significant Item (SSI)	An SI or assembly that is significant due to its requirement to carry high aerodynamic, ground, pressure, or control loads, and whose failure could affect the structural integrity necessary for Product safety, human safety, legal operation or environmental integrity. Refer to S4000P.
substitute part	A part that can replace another part in a specific use. The ability to use a part as a substitute is context-dependent, and is not necessarily a form, fit and function equivalent. – Refer to form, fit and function
supply chain	All organizations directly or indirectly involved in fulfilling a customer support requirement.
supply chain management	The process of planning, implementing and controlling the operations of the supply chain as efficiently as possible.
supply support	Methods, practices and procedures used to determine the requirements of goods and services, as well as their acquisition, receipt, storage, issuance, and final disposal.
support equipment	Equipment required to support the operation and maintenance of a Product.
supportability	The measure of the extent to which all resources required to operate and maintain the Product are properly designed and provided in sufficient quantity.

Term	Definition
supporting task	A supporting task is a part of a complete rectifying or operational support task. Supporting tasks alone cannot rectify issues such as failures, damages, special events or thresholds.
symptom	Evidence of an abnormality directly or indirectly related to the presence of a fault. – A functional symptom may be observed during a functional check and/or by observing the failure of an item associated with the function. – A physical symptom may be observed by visual inspection, measurement of a wear-out parameter, material degradation or by other means. It is detectable and/or measurable regardless of the state of operation of the system.
system breakdown	A system breakdown refers to the partitioning of a Product into a set of systems and subsystems from an engineering point of view. – Refer to functional breakdown
system of systems	A collection of Products, each capable of independent operation, that operate together to enable more advanced capabilities.
task requirement	The determination that a task is necessary to meet supportability requirements.
task resource	Any item necessary to complete a task, such as personnel, spare parts, consumables, raw material, support equipment, facilities or documents.
technical data	All data describing the handling, maintenance, functionality and architecture of a Product.
testability	A design characteristic that allows the status of an item (operable, inoperable, or degraded) to be determined, and the location of any faults and/or failures within the item with a high degree of certainty and in a timely manner.
test equipment	Equipment used to enable or facilitate troubleshooting. – Refer to Built-in Test Equipment (BITE), General Purpose Test Equipment (GPTE), special test equipment
Training Needs Analysis (TNA)	The process used to identify the training requirements for operational and maintenance activities.
Total Cost of Ownership (TCO)	All elements that are part of the LCC, plus the indirect, fixed, and linked costs, such as common support equipment, common facilities, personnel required for unit command, administration, supervision, operations planning and control, fuel, and munitions handling. The TCO represents all costs associated with the ownership of a system, except for non-linked fixed costs that are related to the running of the organization. – Refer to Life Cycle Cost (LCC), Whole Life Cost (WLC)

Term	Definition
toxic substance	– Refer to dangerous substance
troubleshooting	The process of localizing failed replaceable units when their failure is not obvious or localized by other means such as through a built-in test. Troubleshooting is carried out after a failure has been detected.
Usable On Code (UOC)	Analogous to a model identifier in several Logistics Product Data Specifications, for example, refer to GEIA-STD-0007. – Refer to product variant
Whole Life Cost (WLC)	All elements that are a part of the TCO, plus indirect, fixed, and non-linked costs, such as family housing, medical services, ceremonial units, basic training, headquarters and staff, academies and recruiters. – Refer to Life Cycle Cost (LCC), Total Cost of Ownership (TCO)

3 General considerations for the list of abbreviations and acronyms

Whenever in doubt regarding the intelligibility of an abbreviation or acronym, and whenever there is ample space in the document, write the term in full.

3.1 Word combination - acronym

Abbreviations and acronyms that represent combinations of words must be used as shown unless otherwise authorized. Individual parts of abbreviations or acronyms must not be used independently unless those parts are defined as acronyms themselves. Acronyms may be combined if necessary, even if there is no acronym listed for the combination.

3.2 Tense and number

Abbreviations and acronyms are used consistently throughout this specification to represent all instances of the relevant term, whether it is used in the singular or in the plural form.

3.3 Abbreviation and acronym list

Table 3 provides the definitions of abbreviations and acronyms used throughout this specification.

Table 3 Abbreviations and acronyms

Abbreviation / acronym	Definition
A	Availability
acid	Acidification
AIA	Aerospace Industries Association of America
ALC	Alternate Logistics Support Analysis Control Number

Abbreviation / acronym	Definition
AOR	Annual Operating Requirements
ASD	AeroSpace and Defence Industries Association of Europe
ATA	Air Transport Association of America
BE	Breakdown Element
BEI	Breakdown Element Identifier
BER	Breakdown Element Revision
BIT	Built-In Test
BITE	Built-In Test Equipment
BOM	Bill of Material
BPR	Biocidal Products Regulations
BR	Breakdown Revision
CAD	Computer-Aided Design
CAGE	Commercial and Government Entity
CAS	Chemical Abstracts Service
CAT	Substance Category
CBS	Cost Breakdown Structure
CDR	Critical Design Review
CDRL	Contracted Data Requirement List
CFC	Chlorofluorocarbon
Chap	Chapter
CI	Candidate Item
CIL	Candidate Item List
CLP	Classification, Labeling, and Packaging
cm	Centimeter
CM	Configuration Management
CMR	Certification Maintenance Requirement
CONOPS	Concept of Operations
COTS	Commercial Off The Shelf
CRD	Customer Requirements Document
CSC	Computer Software Configuration

Abbreviation / acronym	Definition
CSCI	Computer Software Configuration Item
DA	Design Authority
DADD	Data and Assumptions Definition Document
DEL	Data Element List
DEX	Data Exchange Specification
DLM	Depot Level Maintenance
DMC	Data Module Code
DMS	Diminishing Manufacturing Sources
DMSMS	Diminishing Manufacturing Sources and Material Shortages
DoD	Department of Defense
DRACAS	Data Reporting and Corrective Action System
EBOM	Engineering Bill of Material
EC	European Community
EIAC	End Item Acronym Code
ELORA	Economical Level of Repair Analysis
ELV	End-of Life Vehicle
ene	Energy regaining by burning
FFE	Functional Failure Effect
FFEC	Functional Failure Effect Code
Fig	Figure
FLS	Field Loadable Software
FMA	Failure Modes Analysis
FMDR	Failure Mode Distribution Ratio
FMEA	Failure Modes and Effects Analysis
FMECA	Failure Modes, Effects and Criticality Analysis
FMG	Failure Mode Group
FMR	Failure Mode Ratio
FPGA	Field Programmable Gate Array
ft	foot/feet
FTA	Fault Tree Analysis

Abbreviation / acronym	Definition
GC	Guidance Conference
GD	Guidance Document
GEIA	Government Electronic and Information Technology Association
GFE	Government Furnished Equipment
GPTE	General Purpose Test Equipment
GSE	Ground Support Equipment
GVI	General Visual Inspection
gwp	Greenhouse effect
HCFC	Hydro chlorofluorocarbon
HFES	Human Factors and Ergonomics Society
HIRF	High Intensity Radiated Field
HUMS	Health and Usage Monitoring System
HVAC	Heating, Ventilation, and Air Conditioning
HW	Hardware
IAEA	International Atomic Energy Agency
IATA	International Air Transport Association
ICN	Information Control Number
ILC	Item Location Code
ILS	Integrated Logistics Support
IMO	International Maritime Organization
IPS	Integrated Product Support
ISMO	In-Service Maintenance Optimization
ISO	International Organization for Standardization
ISSO	In-Service Support Optimization
IT	Information Technology
IUA	Item Under Analysis
kg	kilogram
KPI	Key Performance Indicator
LAN	Local Area Network
lb	pound

Abbreviation / acronym	Definition
LCC	Life Cycle Cost
LCCBS	Life Cycle Cost Breakdown Structure
LCN	Logistics Support Analysis Control Number
LORA	Level of Repair Analysis
LRU	Line Replaceable Unit
LSA	Logistics Support Analysis
LSA GC	LSA Guidance Conference
LSA PP	LSA Program Plan
LSA RC	LSA Review Conference
LSAR	Logistics Support Analysis Record
m	Meter
MDT	Mean Maintenance Down Time
MET	Mean Elapsed Time
ML	Maintenance Level
MoD	Ministry of Defence
MoU	Memorandum of Understanding
mr	Material recycling
MRI	Maintenance Relevant Item
MRO	Maintenance, Repair and Overhaul
MSDS	Material Safety Data Sheet
MTA	Maintenance Task Analysis
MTBF	Mean Time Between Failures
MTBM	Mean Time Between Maintenance
MTBUR	Mean Time Between Unscheduled Removal
MTTF	Mean Time To Failure
MTTR	Mean Time To Repair
mw	Material waste
NATO	North Atlantic Treaty Organization
NEA	Nuclear Energy Agency
NFF	No Fault Found

Abbreviation / acronym	Definition
O&S	Operating and Support
OECD	Organization for Economic Co-operation and Development
OEM	Original Equipment Manufacturer
OH	Operating Hour
OM	Obsolescence Management
OMP	Operator Maintenance Plan
ORD	Operational Requirements Document
ozon	Dangerous for ozone layer
Para	Paragraph
PBS	Product Breakdown Structure
PDF	Product Disposal File
PDM	Product Data Management / Product Design Management
PDR	Preliminary Design Review
PE	Planned Event
PHST	Packaging, Handling, Storage and Transportation
PI	Part Identifier
PLCS	Product Life Cycle Support
PLM	Product Lifecycle Management
PMA	Preventive Maintenance Analysis
PMP	Preventive Maintenance Program / Preventive Maintenance Plan
PMTR	Preventative Maintenance Task Requirement
PMTRE	Preventive Maintenance Task Requirement Event
PMTRI	Preventive Maintenance Task Requirement Interval
POP	Persistent Organic Pollutants
pos	Environmental aspects, positive
PP	Program Plan
PSA	Product Support Analysis
RAMT	Reliability, Availability, Maintainability and Testability
RC	Review Conference
RCM	Reliability Centered Maintenance

Abbreviation / acronym	Definition
REACH	Registration, Evaluation, Authorisation and Restriction of Chemicals
RFP	Request for Proposal
RMT	Reliability, Maintainability, Testability
ROHS	Restriction of (the use of certain) Hazardous Substances
ROM	Read-Only Memory
SBOM	Support Bill of Material
SD	Significant Detail / Structural Detail
SDS	Safety Data Sheet
SI	Structural Item
SLA	Special Libraries Association
SLS	Shop Loadable Software
SMP	Scheduled Maintenance Program
SMR	Source, Maintenance, and Recoverability
SNS	Standard Numbering System
SOW	Statement of Work
SRU	Shop Replaceable Unit
SSA	Software Support Analysis
SSC	Software Support Concept
SSI	Structural Significant Item
SW	Software
TBD	To Be Defined
TCO	Total Cost of Ownership
TF	Task Frequency
TNA	Training Needs Analysis
tox	Toxic, bioaccumulation, and/or difficult to decompose
UN	United Nations
UOC	Usable On Code
UoF	Unit of Functionality
US	United States
WAN	Wide Area Network

Abbreviation / acronym	Definition
WBS	Work Breakdown Structure
WEEE	Waste from Electrical and Electronic Equipment
WLC	Whole-Life Cost
WRA	Weapon Replacement Assembly
XML	Extensible Mark-up Language

Chapter 22

Data element list

Table of contents

List of tables

References

Table 1 References

Chap No./Document No.	Title
Chap 19	Data model
S1000D	International specification for technical publications using a common source database
S4000P	International specification for developing and continuously improving preventive maintenance
SX001G	Glossary for the S-Series ILS specifications
SX004G	Unified Modeling Language (UML) model reader's guide
SX005G	S-Series IPS specifications XML schema implementation guidance
ISO 3166-1	Codes for the representation of names of countries and their subdivisions – Part 1: Country codes
ISO 639:1988	Language codes
SAE-GEIA-STD-0007	Logistics Product Data (LPD)

1 General

This chapter defines all data elements (classes, interfaces and attributes) and valid values that are used in the S3000L data model, refer to Chap 19.

2 Classes

The full list of S3000L classes and interfaces is provided in Table 2.

Table 2 is organized alphabetically by class and interface name, and contains:

- Class/interface name
- Class/interface type (refer to SX004G on more details on class and interfaces used in the S-Series IPS specifications
- Class/interface stereotype (refer to SX004G on more details on stereotypes used in the S-Series IPS specifications)
- Class/interface definition
- Unit of Functionality (UoF) identifies the section in Chap 19 where the class or interface is defined

Table 2 List of classes and interfaces

Class name	Type	Stereotype	Definition	UoF
AdditionalTrainingNeed	Class	attributeGroup	AdditionalTrainingNeed is an <<attributeGroup>> that specifies additional learning required for the associated CompetenceDefinitionItem in order to qualify as the TaskPersonnelResource.	S3000L UoF Task Resource
AggregatedElement	Class	class	AggregatedElement is a BreakdownElement that is a container for a collection of BreakdownElements which are grouped for an identified purpose.	S3000L UoF Aggregated Element
AggregatedElementRevision	Class	class	AggregatedElementRevision is a BreakdownElementRevision representing an iteration applied to an AggregatedElement.	S3000L UoF Aggregated Element
AllocatedTaskLocation	Class	class	AllocatedTaskLocation is a <<class>> that identifies where a Task is to be performed in the context of a given support solution.	S3000L UoF Task Usage
AllowedProductConfiguration	Interface	extend	AllowedProductConfiguration is an <<extend>> interface that provides its associated data model to those classes that must define permitted combinations of hardware and software parts which can or must be installed in specific locations (positions). **Note** One and the same serialized product can adhere to different allowed product configurations over time. **Note** An allowed product configuration can also include associated engineering instructions that must be adhered to during assembly and operation and that demonstrates that a product complies with applicable regulations. **Example** – Applicable regulations may be a type certificate.	S3000L UoF Product Design Configuration
AllowedProductConfigurationByConfigurationIdentifier	Class	class	AllowedProductConfigurationByConfigurationIdentifier is a <<class>> that defines an AllowedProductConfiguration by means other than a part number.	S3000L UoF Product Design Configuration

Class name	Type	Stereotype	Definition	UoF
AllowedProductConfigurationHardwarePartAsDesigned	Class	class	AllowedProductConfigurationHardwarePartAsDesigned is a HardwarePartAsDesigned that is managed as an AllowedProductConfiguration.	S3000L UoF Product Design Configuration
AllowedProductConfigurationItem	Interface	select	AllowedProductConfigurationItem is a <<select>> interface that identifies items which can be selected as an allowed product configuration.	S3000L UoF Product Design Configuration
AllowedProductConfigurationsByConfigurationIdentifier	Class	exchange	AllowedProductConfigurationsByConfigurationIdentifier is a wrapper element that contains all instances of AllowedProductConfigurationByConfigurationIdentifier that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content
AlternatePartAsDesigned	Class	relationship	AlternatePartAsDesigned is a <<relationship>> that defines an alternate PartAsDesigned which can replace the base PartAsDesigned in all its usages ie, it is context independent, and is form, fit, and function equivalent. **Note** A part can have one or more alternate parts. The alternate part is interchangeable with the base part in any/all uses.	S3000L UoF Part Definition
AnalysisActivity	Class	class	AnalysisActivity is a <<class>> that represents the objective for, and outcome of, an analysis carried out for the AnalysisCandidateItem.	S3000L UoF LSA Candidate
AnalysisActivityRevision	Class	class	AnalysisActivityRevision is a <<class>> representing an iteration applied to an AnalysisActivity.	S3000L UoF LSA Candidate
AnalysisCandidateItem	Interface	extend	AnalysisCandidateItem is an <<extend>> interface that provides its associated data model to those classes that can have an associated AnalysisActivity.	S3000L UoF LSA Candidate
AnalysisCandidateItemSelectionData	Class	attributeGroup	AnalysisCandidateItemSelectionData is an <<attributeGroup>> that summarizes decisions made for the AnalysisCandidateItem from a support analysis activities perspective.	S3000L UoF LSA Candidate

Class name	Type	Stereotype	Definition	UoF
ApplicabilityDefinitionData	Class	exchange	ApplicabilityDefinitionData is a wrapper element that contains all instances of ApplicabilityStatement that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content
ApplicabilityStatement	Class	class	ApplicabilityStatement is a <<class>> that defines the situation or situations under which related items are valid.	S3000L UoF Applicability Statement
ApplicabilityStatementItem	Interface	extend	ApplicabilityStatementItem is an <<extend>> interface that provides its associated data model to those classes which can have restricted validity as defined by an associated ApplicabilityStatement.	S3000L UoF Applicability Statement
ApplicableDecisionTreeTemplate	Class	relationship	ApplicableDecisionTreeTemplate is a <<relationship>> that identifies a DecisionTreeTemplate which can be used for the DecisionTreeAnalysisItem.	S3000L UoF Decision Tree Template Definition
AssociatedEnvironmentDefinition	Class	relationship	AssociatedEnvironmentDefinition is a <<relationship>> that associates an EnvironmentDefinitionItem with an EnvironmentDefinition relevant to its existence, operation and/or support.	S3000L UoF Environment Definition
AuthorityDrivenTaskRequirement	Class	attributeGroup	AuthorityDrivenTaskRequirement is an <<attributeGroup>> that collects information on the TaskRequirement, where it is derived from regulations and/or other authoritative sources.	S3000L UoF Task Requirement
AuthorityToOperate	Class	class	AuthorityToOperate is a <<class>> that represents a certification allowing a specific configuration of a product to be put into operation. **Note** Type certificate for an aircraft signifies the airworthiness of its design. **Note** A design change cannot be put into operation without re-certification.	S3000L UoF Product Design Configuration
Breakdown	Class	class	Breakdown is a <<class>> that identifies a specific partitioning of a Product to form a parent-child structure of related instances of BreakdownElement.	S3000L UoF Breakdown Structure

Class name	Type	Stereotype	Definition	UoF
BreakdownElement	Class	class	BreakdownElement is a <<class>> defining a partition of a Product that is used in one or many instances of Breakdown.	S3000L UoF Breakdown Structure
BreakdownElementInZone	Class	relationship	BreakdownElementInZone is a <<relationship>> where a BreakdownElementInZoneItem relates to the ZoneElement where it is located.	S3000L UoF Zone Element
BreakdownElementInZoneItem	Interface	extend	BreakdownElementInZoneItem is an <<extend>> interface that provides its associated data model to those classes that implement it.	S3000L UoF Zone Element
BreakdownElementRevision	Class	class	BreakdownElementRevision is a <<class>> representing an iteration applied to a BreakdownElement.	S3000L UoF Breakdown Structure
BreakdownElementRevisionRelationship	Class	relationship	BreakdownElementRevisionRelationship is a <<relationship>> where one BreakdownElementRevision relates to another BreakdownElement or BreakdownElementRevision.	S3000L UoF Breakdown Structure
BreakdownElementRevisionRelationshipItem	Interface	select	BreakdownElementRevisionRelationshipItem is a <<select>> interface that provides the capability to be associated with a BreakdownElementRevision.	S3000L UoF Breakdown Structure
BreakdownElements	Class	exchange	BreakdownElements is a wrapper element that contains all instances of BreakdownElement that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content
BreakdownElementStructure	Class	relationship	BreakdownElementStructure is a <<relationship>> that establishes a hierarchical structure between two usages of BreakdownElement that belong to the same BreakdownRevision.	S3000L UoF Breakdown Structure
BreakdownElementUsageInBreakdown	Class	class	BreakdownElementUsageInBreakdown is a <<class>> that represents a member of a BreakdownRevision. **Note** A BreakdownElementRevision can belong to multiple BreakdownRevisions.	S3000L UoF Breakdown Structure

Class name	Type	Stereotype	Definition	UoF
BreakdownElementUsageRelationship	Class	relationship	BreakdownElementUsageRelationship is a <<relationship>> where one usage of a BreakdownElement relates to the usage of another BreakdownElement. **Note** Both related instances of BreakdownElementUsageInBreakdown must reside within the same BreakdownRevision. **Example** – Version C of a radio is restricted to the use of software version B in breakdown revision 2.	S3000L UoF Breakdown Structure
BreakdownItem	Interface	extend	BreakdownItem is an <<extend>> interface that provides its associated data model to those classes that implement it.	S3000L UoF Breakdown Structure
BreakdownRevision	Class	class	BreakdownRevision is a <<class>> representing an iteration applied to a Breakdown. **Note** BreakdownRevision is used to document design iterations and not breakdown variants.	S3000L UoF Breakdown Structure
BreakdownRevisionRelationship	Class	relationship	BreakdownRevisionRelationship is a <<relationship>> where one BreakdownRevision relates to another BreakdownRevision.	S3000L UoF Breakdown Structure
ChangeAuthorization	Class	class	ChangeAuthorization is a <<class>> that is the record of the permission to modify product design, its procedures and/or associated product support data.	S3000L UoF Change Information
ChangeAuthorizations	Class	exchange	ChangeAuthorizations is a wrapper element that contains all instances of ChangeAuthorization that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content
ChangeControlledItem	Interface	extend	ChangeControlledItem is an <<extend>> interface that provides its associated data model to those classes that can be affected by a ChangeAuthorization.	S3000L UoF Change Information
ChangeNotification	Class	relationship	ChangeNotification is a <<relationship>> that identifies an item changed due to the associated ChangeAuthorization.	S3000L UoF Change Information

Class name	Type	Stereotype	Definition	UoF
ChangeRequest	Class	class	ChangeRequest is a <<class>> that represents a formal proposal for a modification to a configuration item upon a given baseline. **Note** Typical configuration items are eg, Product, PartAsDesigned, and BreakdownElement.	S3000L UoF Design Change Request
ChangeRequestRationale	Class	relationship	ChangeRequestRationale is a <<relationship>> that associates a ChangeRequest with a ChangeRequestRationaleItem.	S3000L UoF Design Change Request
ChangeRequestRationaleItem	Interface	select	ChangeRequestRationaleItem is a <<select>> interface that identifies items which can support a ChangeRequest.	S3000L UoF Design Change Request
ChangeRequests	Class	exchange	ChangeRequests is a wrapper element that contains all instances of ChangeRequest that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content
ChangeRequestTarget	Class	relationship	ChangeRequestTarget is a <<relationship>> that associates a ChangeRequest with a ChangeRequestTargetItem.	S3000L UoF Design Change Request
ChangeRequestTargetItem	Interface	select	ChangeRequestTargetItem is a <<select>> interface that identifies items which can be the subject for a ChangeRequest.	S3000L UoF Design Change Request
CircuitBreaker	Class	class	CircuitBreaker is a <<class>> that represents an individual circuit breaker identified in the context of a defined Product.	S3000L UoF Circuit Breaker
CircuitBreakerItem	Interface	select	CircuitBreakerItem is a <<select>> interface that identifies items which can represent the referenced circuit breaker.	S3000L UoF Task
CircuitBreakerLocation	Class	relationship	CircuitBreakerLocation is a <<relationship>> that identifies the item on which the CircuitBreaker is placed.	S3000L UoF Circuit Breaker
CircuitBreakerLocationItem	Interface	select	CircuitBreakerLocationItem is a <<select>> interface that identifies items which can be selected in order to determine the location for a CircuitBreaker.	S3000L UoF Circuit Breaker
CircuitBreakers	Class	exchange	CircuitBreakers is a wrapper element that contains all instances of CircuitBreaker that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content

Class name	Type	Stereotype	Definition	UoF
CircuitBreakerSetting	Class	class	CircuitBreakerSetting is a <<class>> that specifies an individual circuit breaker that must be in a specific state.	S3000L UoF Task
CircuitBreakerSettings	Class	class	CircuitBreakerSettings is a <<class>> that identifies a set of circuit breakers that must be set in specific states.	S3000L UoF Task
ClassInstanceAssertItem	Interface	select	ClassInstanceAssertItem is a <<select>> interface that identifies classes from which an instance can be used as the EvaluationByAssertionOfClassInstance assert item.	S3000L UoF Applicability Statement
CompetenceDefinitionItem	Interface	select	CompetenceDefinitionItem is a <<select>> interface that identifies items which define measurable or observable possession of knowledge and skills.	S3000L UoF Competence Definition
CompetencyDefinition	Class	class	CompetencyDefinition is a <<class>> that represents the specification of measurable or observable knowledge, skills, and/or abilities necessary for successful performance by a person in a given context. **Note** Competency definition is broadly defined to include assertions of academic, professional, occupational, vocational and life goals, outcomes, and standards, however labeled.	S3000L UoF Competence Definition
CompetencyDefinitions	Class	exchange	CompetencyDefinitions is a wrapper element that contains all instances of CompetencyDefinition that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content
ConditionDefinitionItem	Interface	select	ConditionDefinitionItem is a <<select>> interface that identifies classes from which an instance can be used as the EvaluationByAssertionOfCondition assert condition.	S3000L UoF Applicability Statement
ConditionInstance	Class	class	ConditionInstance is a <<class>> that defines an individual concept or object having the characteristics of a generic ConditionType. **Example** – Uniquely identified service bulletin	S3000L UoF Applicability Statement

Class name	Type	Stereotype	Definition	UoF
ConditionType	Class	class	ConditionType is a <<class>> that defines a concept or an object that needs to be included in applicability statements where the concept or object is not already explicitly represented in the data model. **Example** – Environmental conditions	S3000L UoF Applicability Statement
ConditionTypeAssertMember	Class	class	ConditionTypeAssertMember is a <<class>> that defines a member for a given ConditionType which can be mapped to a Boolean expression and be evaluated to be either TRUE or FALSE.	S3000L UoF Applicability Statement
ConditionTypes	Class	exchange	ConditionTypes is a wrapper element that contains all instances of ConditionType that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content
ContainedSubstance	Class	relationship	ContainedSubstance is a <<relationship>> that associates a HardwarePartAsDesigned with a contained SubstanceDefinition.	S3000L UoF Part Definition
Contract	Class	class	Contract is a <<class>> that represents a binding agreement between two or more parties.	S3000L UoF Product and Project
ContractedProductVariantAtOperatingLocation	Class	relationship	ContractedProductVariantAtOperatingLocation is a <<relationship>> that identifies an OperatingLocation where a contracted ProductVariant is defined to be operated.	S3000L UoF Product Usage Context
ContractedProductVariantAtOperatingLocationType	Class	relationship	ContractedProductVariantAtOperatingLocationType is a <<relationship>> that identifies an OperatingLocationType where a contracted ProductVariant is defined to be operated.	S3000L UoF Product Usage Context
ContractItemDetails	Class	relationship	ContractItemDetails is a <<relationship>> that identifies an item which is the subject of the Contract.	S3000L UoF Product and Project
ContractParty	Class	relationship	ContractParty is a <<relationship>> that identifies a Contract stakeholder.	S3000L UoF Product and Project

Class name	Type	Stereotype	Definition	UoF
ContractRelationship	Class	relationship	ContractRelationship is a <<relationship>> where one Contract relates to another Contract.	S3000L UoF Product and Project
Contracts	Class	exchange	Contracts is a wrapper element that contains all instances of Contract that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content
CorrectionFactor	Class	attributeGroup	CorrectionFactor is an <<attributeGroup>> that defines a mathematical adjustment of a defined value to account for a specified deviation in a given context.	S3000L UoF Performance Parameter
Countries	Class	exchange	Countries is a wrapper element that contains all instances of Country that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content
Country	Class	class	Country is a self-governing political entity, occupying a particular territory.	S3000L UoF Location
DamageAnalysis	Class	class	DamageAnalysis is an AnalysisActivity that represents the objective for, and outcome of, a damage analysis carried out for the AnalysisCandidateItem.	S3000L UoF Damage Definition
DamageAnalysisRevision	Class	class	DamageAnalysisRevision is an AnalysisActivityRevision representing an iteration applied to a DamageAnalysis.	S3000L UoF Damage Definition
DamageCause	Class	relationship	DamageCause is a <<relationship>> where a DamageDefinition relates to a SpecialEventDefinition that in some way is associated with the DamageDefinition.	S3000L UoF Damage Definition
DamageDefinition	Class	class	DamageDefinition is a <<class>> that represents a loss or reduction of functionality due to external causes or use outside specified limits.	S3000L UoF Damage Definition
DamageImpact	Class	relationship	DamageImpact is a <<relationship>> that defines a consequence resulting from the identified DamageDefinition.	S3000L UoF Damage Definition
DecisionTreeAnalysisItem	Interface	extend	DecisionTreeAnalysisItem is an <<extend>> interface that provides its associated data model to those classes that implement it.	S3000L UoF Decision Tree Template Definition

Class name	Type	Stereotype	Definition	UoF
DecisionTreeTemplate	Class	class	DecisionTreeTemplate is a <<class>> that enables the representation of a decision process including a set of defined steps and binary decisions.	S3000L UoF Decision Tree Template Definition
DecisionTreeTemplateActionDefinition	Class	class	DecisionTreeTemplateActionDefinition is a <<class>> that specifies actions and/or measures that must be taken as part of the decision process.	S3000L UoF Decision Tree Template Definition
DecisionTreeTemplateEndActionDefinition	Class	class	DecisionTreeTemplateEndActionDefinition is a DecisionTreeTemplateActionDefinition that defines an end action for the decision process.	S3000L UoF Decision Tree Template Definition
DecisionTreeTemplateFollowOnItem	Interface	select	DecisionTreeTemplateFollowOnItem is a <<select>> interface that identifies items which can be selected to define the next step in the decision process.	S3000L UoF Decision Tree Template Definition
DecisionTreeTemplateFurtherAnalysisActionDefinition	Class	class	DecisionTreeTemplateFurtherAnalysisActionDefinition is a DecisionTreeTemplateActionDefinition that will be followed by another question or action.	S3000L UoF Decision Tree Template Definition
DecisionTreeTemplateQuestionDefinition	Class	class	DecisionTreeTemplateQuestionDefinition is a <<class>> that specifies a question that results in a YES or NO answer.	S3000L UoF Decision Tree Template Definition
DecisionTreeTemplateRevision	Class	class	DecisionTreeTemplateRevision is a <<class>> representing an iteration applied to a DecisionTreeTemplate.	S3000L UoF Decision Tree Template Definition
DecisionTreeTemplates	Class	exchange	DecisionTreeTemplates is a wrapper element that contains all instances of DecisionTreeTemplate that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content
DecisionTreeTemplateStartItem	Interface	select	DecisionTreeTemplateStartItem is a <<select>> interface that identifies items which can be selected as the starting point for the decision process.	S3000L UoF Decision Tree Template Definition
DetectionMeansAlarm	Class	class	DetectionMeansAlarm is a <<class>> that supports the definition and description of an alarm being implemented by a DetectionMeansAlarmItem.	S3000L UoF Failure Mode Symptom

Class name	Type	Stereotype	Definition	UoF
DetectionMeansAlarmItem	Interface	extend	DetectionMeansAlarmItem is an <<extend>> interface that provides its associated data model to those classes that can have an associated DetectionMeansAlarm.	S3000L UoF Failure Mode Symptom
DigitalFile	Class	class	DigitalFile is a <<class>> that provides the identification of data stored on an electronic device that can be interpreted by a computer.	S3000L UoF Digital File
DigitalFileReference	Class	relationship	DigitalFileReference is a <<relationship>> that allows a DigitalFile to reference a DigitalFileReferencedItem.	S3000L UoF Digital File
DigitalFileReferencedItem	Interface	select	DigitalFileReferencedItem is a <<select>> interface that identifies an item which in some way is associated with the content in the DigitalFile.	S3000L UoF Digital File
DigitalFileReferencingItem	Interface	extend	DigitalFileReferencingItem is an <<extend>> interface that provides its associated data model to those classes that implement it.	S3000L UoF Digital File
DigitalFiles	Class	exchange	DigitalFiles is a wrapper element that contains all instances of DigitalFile that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content
DiscreteTimeLimit	Class	class	DiscreteTimeLimit is a TimeLimit that is distinct, where its next possible occurrence cannot be scheduled. **Note** A Trigger is something that activates a DiscreteTimeLimit. **Note** A Threshold is a point that must not be exceeded once the DiscreteTimeLimit has been activated. If there is no threshold value, then the TimeLimitItem must be initiated immediately after the DiscreteTimeLimit activation.	S3000L UoF Time Limit

Class name	Type	Stereotype	Definition	UoF
Document	Class	class	Document is a <<class>> that represents a compiled set of information that serves a purpose. **Example** – Drawing – Report – Manual	S3000L UoF Document
DocumentData	Class	exchange	DocumentData is a wrapper element that contains all instances of Document that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content
DocumentIssue	Class	class	DocumentIssue is a <<class>> that represents a specific release of a Document.	S3000L UoF Document
DocumentItem	Interface	select	DocumentItem is a <<select>> interface that identifies items which can be selected as Document.	S3000L UoF Document
DocumentReferencingItem	Interface	extend	DocumentReferencingItem is an <<extend>> interface that provides its associated data model to those classes that implement it.	S3000L UoF Document
EffectiveOnProductConfiguration	Class	relationship	EffectiveOnProductConfiguration is a <<relationship>> that identifies that an EffectiveOnProductConfigurationItem, included in the Breakdown for the overall Product, is effective in the associated AllowedProductConfiguration.	S3000L UoF Product Design Configuration
EffectiveOnProductConfigurationItem	Interface	extend	EffectiveOnProductConfigurationItem is an <<extend>> interface that provides its associated data model to those classes that can be included in one or many instances of AllowedProductConfiguration.	S3000L UoF Product Design Configuration
EnvironmentDefinition	Class	class	EnvironmentDefinition is a <<class>> that specifies the circumstances, objects, events and/or conditions by which something can be surrounded and that influence the performance of an associated item.	S3000L UoF Environment Definition

Class name	Type	Stereotype	Definition	UoF
EnvironmentDefinitionCharacteristic	Class	class	EnvironmentDefinitionCharacteristic is a <<class>> that represents a measurable or observable characteristic for a circumstance, object, event or condition that is significant to the EnvironmentDefinition.	S3000L UoF Environment Definition
EnvironmentDefinitionItem	Interface	extend	EnvironmentDefinitionItem is an <<extend>> interface that provides its associated data model to those classes that implement it.	S3000L UoF Environment Definition
EnvironmentDefinitionRelationship	Class	relationship	EnvironmentDefinitionRelationship is a <<relationship>> where one EnvironmentDefinition relates to another EnvironmentDefinition.	S3000L UoF Environment Definition
EnvironmentDefinitionRevision	Class	class	EnvironmentDefinitionRevision is a <<class>> representing an iteration applied to an EnvironmentDefinition.	S3000L UoF Environment Definition
EnvironmentDefinitions	Class	exchange	EnvironmentDefinitions is a wrapper element that contains all instances of EnvironmentDefinition that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content
EvaluationByAssertionOfClassInstance	Class	class	EvaluationByAssertionOfClassInstance is an EvaluationCriteria that identifies a class instance to be used as an assert item and be mapped to a Boolean expression which can be evaluated to be either TRUE or FALSE.	S3000L UoF Applicability Statement
EvaluationByAssertionOfCondition	Class	class	EvaluationByAssertionOfCondition is an EvaluationCriteria that identifies a combination of a defined condition and a defined value to be used as an assert item and be mapped to a Boolean expression which can be evaluated to be either TRUE or FALSE.	S3000L UoF Applicability Statement
EvaluationByAssertionOfSerializedItems	Class	class	EvaluationByAssertionOfSerializedItems is an EvaluationCriteria that identifies a class instance together with an associated serial number range to be used as an assert item and be mapped to a Boolean expression which can be evaluated to be either TRUE or FALSE.	S3000L UoF Applicability Statement

Class name	Type	Stereotype	Definition	UoF
EvaluationByNestedApplicabilityStatement	Class	class	EvaluationByNestedApplicabilityStatement is an EvaluationCriteria that enables an ApplicabilityStatement to be reused as part of this EvaluationCriteria. **Note** This class enables the definition of nested applicability statements.	S3000L UoF Applicability Statement
EvaluationCriteria	Class	class	EvaluationCriteria is a <<class>> that defines conditions that can be mapped to a Boolean expression which can be evaluated to be either TRUE or FALSE.	S3000L UoF Applicability Statement
EventThresholdDefinition	Class	class	EventThresholdDefinition is a ThresholdDefinition that is driven by occurrences of related TimeLimitEventItems.	S3000L UoF Time Limit
Facilities	Class	exchange	Facilities is a wrapper element that contains all instances of Facility that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content
Facility	Class	class	Facility is a <<class> that represents a physically limited infrastructure which exists, or is intended to be built or installed, and is established to serve a particular purpose.	S3000L UoF Facility
FacilityRelationship	Class	relationship	FacilityRelationship is a <<relationship>> where one Facility relates to another Facility.	S3000L UoF Facility
FailureMode	Class	class	FailureMode is a <<class>> that defines a functional consequence of an unacceptable state of the FailureModeAnalysisItem. **Example** – No output from electrical circuit.	S3000L UoF Failure Mode
FailureModeAnalysis	Class	class	FailureModeAnalysis is a <<class>> that represents failure modes, effects and criticality identified for the associated FailureModeAnalysisItem.	S3000L UoF Failure Mode
FailureModeAnalysisItem	Interface	extend	FailureModeAnalysisItem is an <<extend>> interface that provides its associated data model to those classes that can have an associated FailureModeAnalysis.	S3000L UoF Failure Mode

Class name	Type	Stereotype	Definition	UoF
FailureModeAnalysisRevision	Class	class	FailureModeAnalysisRevision is a <<class>> representing an iteration applied to a FailureModeAnalysis.	S3000L UoF Failure Mode
FailureModeCause	Class	class	FailureModeCause is a <<class>> that specifies the physical or chemical process(es) that is the reason for the FailureMode.	S3000L UoF Failure Mode
FailureModeCauseItem	Interface	select	FailureModeCauseItem is a <<select>> interface that identifies items which can be selected as being associated with a FailureModeCause.	S3000L UoF Failure Mode
FailureModeCauseItemRelationship	Class	relationship	FailureModeCauseItemRelationship is a <<relationship>> where a FailureModeCause relates to the FailureModeCauseItem that in some way is associated with the FailureModeCause.	S3000L UoF Failure Mode
FailureModeEffect	Class	class	FailureModeEffect is a <<class>> that defines the consequences of an identified FailureMode on the operation, function, or status for the referred item.	S3000L UoF Failure Mode
FailureModeEffectItem	Interface	select	FailureModeEffectItem is a <<select>> interface that identifies items which can be selected as being affected by a FailureMode.	S3000L UoF Failure Mode
FailureModeEffectItemRelationship	Class	relationship	FailureModeEffectItemRelationship is a <<relationship>> where a FailureModeEffect relates to the FailureModeEffectItem that is affected by the FailureMode.	S3000L UoF Failure Mode
FailureModeIsolationCharacteristics	Class	attributeGroup	FailureModeIsolationCharacteristics is an <<attributeGroup>> that collects data which describes the ability to determine that it is the associated failure mode that has occurred.	S3000L UoF Failure Mode Isolation
FailureModeIsolationItem	Interface	extend	FailureModeIsolationItem is an <<extend>> interface that provides its associated data model to those classes that implement it.	S3000L UoF Failure Mode Isolation
FailureModeSymptom	Class	relationship	FailureModeSymptom is a <<relationship>> that identifies a measurable or visible parameter whose appearance can be related, directly or indirectly, to the occurrence of the associated failure mode.	S3000L UoF Failure Mode Symptom

Class name	Type	Stereotype	Definition	UoF
FailureModeSymptomItem	Interface	select	FailureModeSymptomItem is a <<select>> interface that identifies items which can provide a measurable or visible parameter whose appearance can be related, directly or indirectly, to the occurrence of the associated failure mode.	S3000L UoF Failure Mode Symptom
FailureModeSymptomsSignature	Class	class	FailureModeSymptomsSignature is a <<class>> that identifies a set of failure mode symptoms and effects which can be associated with a failure mode. **Note** Failure mode symptoms can be human observable as well as alarms and measurements.	S3000L UoF Failure Mode Symptom
FailureModeSymptomsSignatureItem	Interface	extend	FailureModeSymptomsSignatureItem is an <<extend>> interface that provides its associated data model to those classes that can have an associated FailureModeSymptomsSignature.	S3000L UoF Failure Mode Symptom
FinalInServiceOptimizationAnalysisStep	Class	class	FinalInServiceOptimizationAnalysisStep is an InServiceOptimizationAnalysisStep that specifies final actions and measures taken together with the decision process conclusion and recommendations.	S3000L UoF In Service Optimization Analysis
FollowOnInServiceOptimizationAnalysis	Class	relationship	FollowOnInServiceOptimizationAnalysis is a <<relationship>> where the outcome from the InServiceOptimizationAnalysis resulted in an additional InServiceOptimizationAnalysis.	S3000L UoF In Service Optimization Analysis
FurtherAnalysisInServiceOptimizationAnalysisStep	Class	class	FurtherAnalysisInServiceOptimizationAnalysisStep is an InServiceOptimizationAnalysisStep that specifies actions and measures taken before continuing to the next step defined in the associated DecisionTreeTemplate.	S3000L UoF In Service Optimization Analysis
GeographicalArea	Class	class	GeographicalArea is a <<class>> that represents a particular extent of space.	S3000L UoF Location
GeographicalAreas	Class	exchange	GeographicalAreas is a wrapper element that contains all instances of GeographicalArea that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content

Class name	Type	Stereotype	Definition	UoF
GlobalPosition	Class	class	GlobalPosition is a <<class>> that identifies a point in space by a set of coordinates.	S3000L UoF Location
GlobalPositions	Class	exchange	GlobalPositions is a wrapper element that contains all instances of GlobalPosition that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content
HardwareElement	Class	class	HardwareElement is a BreakdownElement that is realized as a HardwarePartAsDesigned.	S3000L UoF Hardware Element
HardwareElementPartRealization	Class	relationship	HardwareElementPartRealization is a <<relationship>> where a HardwareElementRevision relates to an instance of HardwarePartAsDesigned which fulfills the HardwareElement specification.	S3000L UoF Hardware Element
HardwareElementRevision	Class	class	HardwareElementRevision is a BreakdownElementRevision representing an iteration applied to a HardwareElement.	S3000L UoF Hardware Element
HardwarePartAsDesigned	Class	class	HardwarePartAsDesigned is a PartAsDesigned that is to be realized as physical items, including non-countable material. **Note** Examples of non-countable materials are: oil, sealant, paint.	S3000L UoF Part Definition
HardwarePartAsDesignedDesignData	Class	attributeGroup	HardwarePartAsDesignedDesignData is an <<attributeGroup>> that collects HardwarePartAsDesigned characteristics identified during design activities.	S3000L UoF Part Definition
HardwarePartAsDesignedSupportData	Class	attributeGroup	HardwarePartAsDesignedSupportData is an <<attributeGroup>> that collects HardwarePartAsDesigned characteristics identified during supportability analysis activities.	S3000L UoF Part Definition
IdentifiedTaskRequirement	Class	relationship	IdentifiedTaskRequirement is a <<relationship>> that associates a TaskRequirement with a TaskRequirementAnalysisItem.	S3000L UoF Task Requirement

Class name	Type	Stereotype	Definition	UoF
InfrastructureCompliance	Class	relationship	InfrastructureCompliance is a <<relationship>> that documents how the InfrastructureCompliantItem fulfills requirements stated in the associated ResourceSpecification.	S3000L UoF Facility
InfrastructureCompliantItem	Interface	extend	InfrastructureCompliantItem is an <<extend>> interface that provides its associated data model to those classes that implement it.	S3000L UoF Facility
InServiceOptimizationAnalysis	Class	class	InServiceOptimizationAnalysis is a <<class>> that represents the result from an in-service optimization analysis carried out for the InServiceOptimizationAnalysisItem.	S3000L UoF In Service Optimization Analysis
InServiceOptimizationAnalysisItem	Interface	extend	InServiceOptimizationAnalysisItem is an <<extend>> interface that provides its associated data model to those classes that can have an associated InServiceOptimizationAnalysis.	S3000L UoF In Service Optimization Analysis
InServiceOptimizationAnalysisRevision	Class	class	InServiceOptimizationAnalysisRevision is a <<class>> representing an iteration applied to an InServiceOptimizationAnalysis.	S3000L UoF In Service Optimization Analysis
InServiceOptimizationAnalysisStep	Class	class	InServiceOptimizationAnalysisStep is a <<class>> that represents the results from an individual step in the associated decision process.	S3000L UoF In Service Optimization Analysis
Location	Interface	extend	Location is an <<extend>> interface that provides its associated data model to those classes that implement it.	S3000L UoF Location
LocationItem	Interface	select	LocationItem is a <<select>> interface that identifies items which can be selected to provide the definition of a geographic location.	S3000L UoF Location
LocationRelationship	Class	relationship	LocationRelationship is a <<relationship>> where one LocationItem relates to another LocationItem.	S3000L UoF Location
LogicalAND	Class	class	LogicalAND is an EvaluationCriteria that defines a Boolean operation where the results of all its associated EvaluationCriteria must be TRUE for the result to be TRUE, otherwise the result is FALSE.	S3000L UoF Applicability Statement

Class name	Type	Stereotype	Definition	UoF
LogicalNOT	Class	class	LogicalNOT is an EvaluationCriteria that defines a Boolean operation where the result from its associated EvaluationCriteria must be FALSE for the result to be TRUE, otherwise the result is FALSE.	S3000L UoF Applicability Statement
LogicalOR	Class	class	LogicalOR is an EvaluationCriteria that defines a Boolean operation where the result from at least one of its associated EvaluationCriteria must be TRUE for the result to be TRUE, otherwise the result is FALSE.	S3000L UoF Applicability Statement
LogicalXOR	Class	class	LogicalXOR is an EvaluationCriteria that defines a Boolean operation where the result from one and only one of its associated EvaluationCriteria must be TRUE for the result to be TRUE, otherwise the result is FALSE.	S3000L UoF Applicability Statement
LogisticsSupportAnalysisMessageContent	Class	exchange	LogisticsSupportAnalysisMessageContent is a MessageContent that contains data resulting from LSA activities.	S3000L UoF Logistics Support Analysis Message Content
LogisticsSupportAnalysisPrimaryData	Class	exchange	LogisticsSupportAnalysisPrimaryData is a subset of the LogisticsSupportAnalysisMessageContent <<exchange>> dataset and include the primary data used in and resulting from the LSA activities.	S3000L UoF Logistics Support Analysis Message Content
LogisticsSupportAnalysisSupportingData	Class	exchange	LogisticsSupportAnalysisSupportingData is a subset of the LogisticsSupportAnalysisMessageContent <<exchange>> dataset and include data that support the LSA activities.	S3000L UoF Logistics Support Analysis Message Content
LSACorrectiveMaintenanceAnalysis	Class	class	LSACorrectiveMaintenanceAnalysis is an AnalysisActivity that represents the outcome of corrective maintenance analysis carried out for the AnalysisCandidateItem.	S3000L UoF LSA Failure Mode Group
LSACorrectiveMaintenanceAnalysisRevision	Class	class	LSACorrectiveMaintenanceAnalysisRevision is an AnalysisActivityRevision representing an iteration applied to LSACorrectiveMaintenanceAnalysis.	S3000L UoF LSA Failure Mode Group

Class name	Type	Stereotype	Definition	UoF
LSACorrectiveMaintenanceAnalysisSource	Class	relationship	LSACorrectiveMaintenanceAnalysisSource is a <<relationship>> where an LSACorrectiveMaintenanceAnalysisRevision relates to the failure mode related source information which is used to identify a required corrective maintenance task.	S3000L UoF LSA Failure Mode Group
LSACorrectiveMaintenanceAnalysisSourceItem	Interface	select	LSACorrectiveMaintenanceAnalysisSourceItem is a <<select>> interface that identifies the failure mode source information used to define the required corrective maintenance task.	S3000L UoF LSA Failure Mode Group
LSAFailureModeGroup	Class	class	LSAFailureModeGroup is a <<class>> that represents a set of failure modes that leads to the same set of actions for its detection, isolation and rectifying intervention.	S3000L UoF LSA Failure Mode Group
LSAFailureModeGroupIncludedFailureMode	Class	relationship	LSAFailureModeGroupIncludedFailureMode is a <<relationship>> that identifies a FailureMode which is a member of the LSAFailureModeGroup.	S3000L UoF LSA Failure Mode Group
LSAFailureModeGroupItem	Interface	select	LSAFailureModeGroupItem is a <<select>> interface that identifies failure modes which are included in the LSAFailureModeGroup.	S3000L UoF LSA Failure Mode Group
LSAFailureModeGroupTaskRequirement	Class	relationship	LSAFailureModeGroupTaskRequirement is a <<relationship>> where an LSAFailureModeGroup relates to the TaskRequirement needed in the process to either detect and isolate a failure, or to restore the AnalysisCandidateItem.	S3000L UoF LSA Failure Mode Group
MaintenanceCapabilityAtOperatingLocationType	Class	relationship	MaintenanceCapabilityAtOperatingLocationType is a <<relationship>> that identifies maintenance capabilities which are available at the defined OperatingLocationType.	S3000L UoF Product Usage Context
MaintenanceFacility	Class	class	MaintenanceFacility is a Facility that is mainly established for providing product support.	S3000L UoF Facility

Class name	Type	Stereotype	Definition	UoF
MaintenanceLevel	Class	class	MaintenanceLevel is a <<class>> that represents the definition of a set of maintenance capabilities which will be made available to support a defined Product. **Note** MaintenanceLevel might be established either by a single organization or be distributed between a set of organizations.	S3000L UoF Product Usage Context
MaintenanceLevels	Class	exchange	MaintenanceLevels is a wrapper element that contains all instances of MaintenanceLevel that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content
MeasurementPointDefinition	Class	class	MeasurementPointDefinition is a <<class>> that specifies a characteristic which can be observed for a MeasurementPointDefinitionItem.	S3000L UoF Failure Mode Symptom
MeasurementPointDefinitionItem	Interface	extend	MeasurementPointDefinitionItem is an <<extend>> interface that provides its associated data model to those classes that can have an associated MeasurementPointDefinition.	S3000L UoF Failure Mode Symptom
MeasurementPointDegradedStateDefinition	Class	attributeGroup	MeasurementPointDegradedStateDefinition is an <<attributeGroup>> that specifies a measurable condition which, when met, identifies that a degraded state has been entered.	S3000L UoF Time Limit
MeasurementPointFaultStateDefinition	Class	attributeGroup	MeasurementPointFaultStateDefinition is an <<attributeGroup>> that specifies a measurable condition which, when met, can identify that a failure has occurred.	S3000L UoF Failure Mode Symptom
Message	Class	class	Message is a <<class>> that represents the collection of information brought together by a message sender for the purpose of communicating it to another party.	S3000L UoF Message
MessageContent	Class	exchange	MessageContent is a <<exchange>> definition that represents the collection of information that is the subject of the Message.	S3000L UoF Message

Class name	Type	Stereotype	Definition	UoF
MessageContext	Class	relationship	MessageContext is a <<relationship>> between a Message and the context for which it is being provided. **Example** - Product - Project - Contract	S3000L UoF Message
MessageContextItem	Interface	select	MessageContextItem is a <<select>> interface that identifies items which can be selected as the context for a Message.	S3000L UoF Message
MessageParty	Class	relationship	MessageParty is a <<relationship>> between a Message and a stakeholder for the Message.	S3000L UoF Message
MessagePartyItem	Interface	select	MessagePartyItem is a <<select>> interface that identifies items which can be selected as the party for a Message.	S3000L UoF Message
MessageRelationship	Class	relationship	MessageRelationship is a <<relationship>> where one Message relates to another Message. **Example** - One Message is a reply to another Message - One Message is an update to another Message	S3000L UoF Message
NestedAllowedProductConfiguration	Class	relationship	NestedAllowedProductConfiguration is a <<relationship>> that defines that one AllowedProductConfiguration includes a subordinate AllowedProductConfiguration.	S3000L UoF Product Design Configuration
NestedProductVariant	Class	relationship	NestedProductVariant is a <<relationship>> that defines that one ProductVariant includes a subordinate ProductVariant.	S3000L UoF Product Design Configuration
NonConformanceData	Class	attributeGroup	NonConformanceData is an <<attributeGroup>> that collects information on how the EffectiveOnProductConfigurationItem does not comply with the requirements of its usage.	S3000L UoF Product Design Configuration

Class name	Type	Stereotype	Definition	UoF
OperatingBase	Class	class	OperatingBase is a Facility that is mainly established for providing support for operations.	S3000L UoF Facility
OperatingLocationType	Class	class	OperatingLocationType is a <<class>> that represents the definition of the nature of the environment in which a product will be operated.	S3000L UoF Product Usage Context
OperatingLocationTypes	Class	exchange	OperatingLocationTypes is a wrapper element that contains all instances of OperatingLocationType that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content
OperationalTask	Class	class	OperationalTask is a Task that is required to support the use of a product. **Example** – Fueling – Towing	S3000L UoF Task
Organization	Class	class	Organization is a <<class>> that represents an administrative structure with a particular purpose belonging to a legal entity. **Example** – International agency – Department – Government department – Company	S3000L UoF Organization Assignment
OrganizationReferencingItem	Interface	extend	OrganizationReferencingItem is an <<extend>> interface that provides its associated data model to those classes that implement it.	S3000L UoF Organization Assignment
Organizations	Class	exchange	Organizations is a wrapper element that contains all instances of Organization that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content
ParameterThresholdDefinition	Class	class	ParameterThresholdDefinition is a ThresholdDefinition that is continuously measured and evaluated, and when reached, activates the associated trigger or threshold.	S3000L UoF Time Limit

Class name	Type	Stereotype	Definition	UoF
PartAsDesigned	Class	class	PartAsDesigned is a <<class>> that represents the definitional information for an artifact fulfilling a set of requirements, which can be produced or realized.	S3000L UoF Part Definition
PartAsDesignedControlledItemData	Class	attributeGroup	PartAsDesignedControlledItemData is an <<attributeGroup>> that collects PartAsDesigned characteristics identified during different controlled item analysis activities.	S3000L UoF Part Definition
PartAsDesignedDesignData	Class	attributeGroup	PartAsDesignedDesignData is an <<attributeGroup>> that collects PartAsDesigned characteristics identified during design activities.	S3000L UoF Part Definition
PartAsDesignedPartsList	Class	class	PartAsDesignedPartsList is a <<class>> that represents the definitional information for the collection of PartAsDesignedPartsListEntry included in the assembly of the parent PartAsDesigned. **Note** PartAsDesignedPartsList is typically referred to as a Bill of Material (BOM).	S3000L UoF Part Definition
PartAsDesignedPartsListEntry	Class	class	PartAsDesignedPartsListEntry is a <<class>> that represents the inclusion of a PartAsDesigned in a PartAsDesignedPartsListRevision.	S3000L UoF Part Definition
PartAsDesignedPartsListRelationship	Class	relationship	PartAsDesignedPartsListRelationship is a <<relationship>> where one PartAsDesignedPartsList relates to another PartAsDesignedPartsList.	S3000L UoF Part Definition
PartAsDesignedPartsListRevision	Class	class	PartAsDesignedPartsListRevision is a <<class>> representing an iteration applied to a PartAsDesignedPartsList.	S3000L UoF Part Definition
PartAsDesignedSupportData	Class	attributeGroup	PartAsDesignedSupportData is an <<attributeGroup>> that collects PartAsDesigned characteristics identified during supportability analysis activities.	S3000L UoF Part Definition

Class name	Type	Stereotype	Definition	UoF
Parts	Class	exchange	Parts is a wrapper element that contains all instances of PartAsDesigned that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content
PerformanceParameter	Class	class	PerformanceParameter is a <<class>> that represents a metric that if changed, or not fulfilled, can have a major impact on the performance, schedule, cost and/or risk for the PerformanceParameterItem.	S3000L UoF Performance Parameter
PerformanceParameterItem	Interface	extend	PerformanceParameterItem is an <<extend>> interface that provides its associated data model to those classes that can have an associated PerformanceParameter.	S3000L UoF Performance Parameter
PerformanceParameterRevision	Class	class	PerformanceParameterRevision is a <<class>> that represents an iteration applied to a PerformanceParameter.	S3000L UoF Performance Parameter
PerformanceParameterValueGroup	Class	attributeGroup	PerformanceParameterValueGroup is an <<attributeGroup>> that organizes PerformanceParameter values for a defined purpose.	S3000L UoF Performance Parameter
PeriodicTimeLimit	Class	class	PeriodicTimeLimit is a TimeLimit that is repeated and its next occurrence can be scheduled.	S3000L UoF Time Limit
PlannedTaskItem	Interface	select	PlannedTaskItem is a <<select>> interface that identifies items which can be used in the context of PlannedTaskUsage.	S3000L UoF Task Usage
PlannedTaskUsage	Class	relationship	PlannedTaskUsage is a TaskUsage that expands the definition of a required Task in the context of a given support solution. **Note** Both rectifying and operational tasks are justified by task requirements identified during various analysis activities.	S3000L UoF Task Usage

Class name	Type	Stereotype	Definition	UoF
Product	Class	class	Product is a <<class>> that represents a family of items which share the same underlying design purpose. **Example** – Airbus A340 – iPhone 12 – Pegasus engine – Ford Fusion – Stryker – Aegis Class Destroyer	S3000L UoF Product and Project
Products	Class	exchange	Products is a wrapper element that contains all instances of Product that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content
ProductUsagePhase	Class	class	ProductUsagePhase is a <<class>> that represents a distinct period of time during which a ProductUsagePhaseItem is in an operational state which has specific characteristics that need special considerations. **Example** – Preflight, post flight, cruise, and taxiing for an aircraft. – Emersion, surface and dock for a submarine.	S3000L UoF Product Usage Phase
ProductUsagePhaseItem	Interface	extend	ProductUsagePhaseItem is an <<extend>> interface that provides its associated data model to those classes that implement it.	S3000L UoF Product Usage Phase
ProductUsagePhaseRelationship	Class	relationship	ProductUsagePhaseRelationship is a <<relationship>> where one ProductUsagePhase relates to another ProductUsagePhase.	S3000L UoF Product Usage Phase
ProductVariant	Class	class	ProductVariant is a <<class>> that defines a member of a Product family which is configured for a specific purpose and is made available to the market. **Note** A product variant is often known as a model. **Example** – Boeing 787-800 vs 787-900 – Ford Fusion S vs. SE vs. SEL	S3000L UoF Product and Project

Class name	Type	Stereotype	Definition	UoF
Project	Class	class	Project is a <<class>> that represents the overall set of Integrated Product Support (IPS) activities defined for a Product. **Note** Project is often referred to as an IPS program.	S3000L UoF Product and Project
ProjectContract	Class	relationship	ProjectContract is a <<relationship>> that establishes an association between a Project and a Contract.	S3000L UoF Product and Project
Projects	Class	exchange	Projects is a wrapper element that contains all instances of Project that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content
QuestionInServiceOptimizationAnalysisStep	Class	class	QuestionInServiceOptimizationAnalysisStep is an InServiceOptimizationAnalysisStep that specifies the answer to a question defined in the associated DecisionTreeTemplate.	S3000L UoF In Service Optimization Analysis
RectifyingTask	Class	class	RectifyingTask is a Task that ensures or returns the function of the associated item. **Note** An event that can require a Task to be performed includes a failure, damage, a special event, and a time limit. **Example** – Lubrication – Repair – Replace	S3000L UoF Task
ReferencedDigitalFile	Class	relationship	ReferencedDigitalFile is a <<relationship> that allows an item to refer to a DigitalFile.	S3000L UoF Digital File
ReferencedDocument	Class	relationship	ReferencedDocument is a <<relationship>> where one DocumentReferencingItem relates to a DocumentItem.	S3000L UoF Document
ReferencedOrganization	Class	relationship	ReferencedOrganization is a <<relationship>> where one OrganizationReferencingItem relates to an Organization.	S3000L UoF Organization Assignment

Class name	Type	Stereotype	Definition	UoF
Remark	Class	attributeGroup	Remark is an <<attributeGroup>> that provides additional information about the associated item. **Note** A remark may be a personal opinion (“I prefer more onions in my soup”) or it may be a technical fact (“The manufacturer recommends heating the soup to 45 degrees Celsius”).	S3000L UoF Remark
RemarkItem	Interface	extend	RemarkItem is an <<extend>> interface that provides its associated data model to those classes that implement it.	S3000L UoF Remark
RepeatDefinition	Class	class	RepeatDefinition is a <<class>> that represents the set of circumstances which define the PeriodicTimeLimit. **Note** Only one RepeatDefinition must be active at a specific moment in time. **Note** A trigger is something that activates a RepeatDefinition. **Note** A threshold is a point that must not be exceeded by the TimeLimitItem once the RepeatDefinition has been activated.	S3000L UoF Time Limit
ResourceRealization	Class	relationship	ResourceRealization is a <<relationship>> where a ResourceSpecification relates to an instance of PartAsDesigned that fulfills the resource specification.	S3000L UoF Resource Specification
ResourceSpecification	Class	class	ResourceSpecification is a <<class>> that defines a resource by its characteristics. **Note** ResourceSpecification allows for more generic resource definitions ie, a task/subtask does not need to be changed due to eg, customer specific resource preferences.	S3000L UoF Resource Specification
ResourceSpecificationRevision	Class	class	ResourceSpecificationRevision is a <<class>> representing an iteration applied to a ResourceSpecification.	S3000L UoF Resource Specification

Class name	Type	Stereotype	Definition	UoF
ResourceSpecifications	Class	exchange	ResourceSpecifications is a wrapper element that contains all instances of ResourceSpecification that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content
ResultingDataModule	Class	relationship	ResultingDataModule is a <<relationship>> that identifies a data module issue in which a TaskRevision is further detailed. **Note** A data module can contain either a complete or partial description of a given Task.	S3000L UoF Task
S1000DDataModule	Class	class	S1000DDataModule is a Document that is written in accordance with an S1000D schema.	S3000L UoF Document
S1000DDataModuleIssue	Class	class	S1000DDataModuleIssue is a DocumentIssue that identifies a specific issue of a data module produced in accordance with S1000D.	S3000L UoF Document
S1000DPublicationModule	Class	class	S1000DPublicationModule is a Document that identifies a publication published in accordance with S1000D.	S3000L UoF Document
S1000DPublicationModuleIssue	Class	class	S1000DPublicationModuleIssue is a DocumentIssue that identifies a specific issue of a publication module published in accordance with S1000D.	S3000L UoF Document
SamplingDefinition	Class	attributeGroup	SamplingDefinition is an <<attributeGroup>> that specifies whether the associated action is to be performed on a subset of the population of the associated target item.	S3000L UoF Time Limit
SamplingItem	Interface	extend	SamplingItem is an <<extend>> interface that represents the common behavior for those classes which can have an associated SamplingDefinition.	S3000L UoF Time Limit
SecurityClass	Class	class	SecurityClass is a <<class>> that identifies a level of confidentiality which can be used to protect something against unauthorized access.	S3000L UoF Security Classification
SecurityClasses	Class	exchange	SecurityClasses is a wrapper element that contains all instances of SecurityClass that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content

Class name	Type	Stereotype	Definition	UoF
SecurityClassification	Class	relationship	SecurityClassification is a <<relationship>> that associates a given SecurityClass with the item that must be protected against unauthorized access or distribution.	S3000L UoF Security Classification
SecurityClassificationItem	Interface	extend	SecurityClassificationItem is an <<extend>> interface that provides its associated data model to those classes that implement it.	S3000L UoF Security Classification
SerializedAssertItem	Interface	select	SerializedAssertItem is a <<select>> interface that identifies classes from which an instance can be used as the EvaluationByAssertionOfSerializedItems assert item.	S3000L UoF Applicability Statement
Skill	Class	class	Skill is a <<class>> that represents human cognitive, psychomotor, and affective abilities.	S3000L UoF Competence Definition
SkillLevel	Class	class	SkillLevel is a <<class>> that represents a defined proficiency of a Trade.	S3000L UoF Competence Definition
Skills	Class	exchange	Skills is a wrapper element that contains all instances of Skill that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content
SoftwareElement	Class	class	SoftwareElement is a BreakdownElement that is realized as a SoftwarePartAsDesigned.	S3000L UoF Software Element
SoftwareElementPartRealization	Class	relationship	SoftwareElementPartRealization is a <<relationship>> where a SoftwareElementRevision relates to an instance of SoftwarePartAsDesigned which fulfills the SoftwareElement specification.	S3000L UoF Software Element
SoftwareElementRevision	Class	class	SoftwareElementRevision is a BreakdownElementRevision representing an iteration applied to a SoftwareElement.	S3000L UoF Software Element
SoftwarePartAsDesigned	Class	class	SoftwarePartAsDesigned is a PartAsDesigned that is produced as an executable software or as a data file. **Note** Non-executable software includes eg, maps.	S3000L UoF Part Definition

Class name	Type	Stereotype	Definition	UoF
SoftwarePartAsDesignedDesignData	Class	attributeGroup	SoftwarePartAsDesignedDesignData is an <<attributeGroup>> that collects SoftwarePartAsDesigned characteristics identified during design activities.	S3000L UoF Part Definition
SpecialEventAnalysis	Class	class	SpecialEventAnalysis is an AnalysisActivity that represents the objective for, and outcome of, a special event analysis carried out for the AnalysisCandidateItem.	S3000L UoF Special Event
SpecialEventAnalysisRevision	Class	class	SpecialEventAnalysisRevision is an AnalysisActivityRevision representing an iteration applied to a SpecialEventAnalysis.	S3000L UoF Special Event
SpecialEventDefinition	Class	class	SpecialEventDefinition is a <<class>>that represents a typical happening which can be induced to the item under analysis during its normal operation, and can lead to damages.	S3000L UoF Special Event
SpecialEventProductUsagePhaseOccurrence	Class	relationship	SpecialEventProductUsagePhaseOccurrence is a <<relationship>> that defines an association between an instance of SpecialEventDefinition and an instance of ProductUsagePhase during which the special event can occur.	S3000L UoF Special Event
StreetAddress	Class	class	StreetAddress is a <<class> that represents a locatable position along a road.	S3000L UoF Location
SubstanceDefinition	Class	class	SubstanceDefinition is a <<class>> that identifies high concern physical matter. **Example** – CAS (Chemical Abstract Substance) Registry – REACH (Registration, Evaluation, Authorisation and restriction of Chemicals)	S3000L UoF Part Definition
SubstanceDefinitions	Class	exchange	SubstanceDefinitions is a wrapper element that contains all instances of SubstanceDefinition that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content

Class name	Type	Stereotype	Definition	UoF
SubstitutePartAsDesigned	Class	relationship	SubstitutePartAsDesigned is a <<relationship>> that defines a substitute PartAsDesignedPartsListEntry which can replace the base PartAsDesignedPartsListEntry in the context of the parent PartAsDesignedPartsList.	S3000L UoF Part Definition
Subtask	Class	class	Subtask is a <<class>> that represents the specification of a work step that is to be performed as part of a Task.	S3000L UoF Task
SubtaskAcceptanceParameter	Class	class	SubtaskAcceptanceParameter is a <<class>> that represents acceptance criteria which must be met before the Subtask is completed.	S3000L UoF Task
SubtaskByDefinition	Class	class	SubtaskByDefinition is a Subtask that provides detailed information of the defined work step.	S3000L UoF Task
SubtaskByTaskReference	Class	class	SubtaskByTaskReference is a Subtask where the details of the subtask are defined as a separate Task.	S3000L UoF Task
SubtaskInZone	Class	relationship	SubtaskInZone is a <<relationship>> that identifies the zone where the Subtask is to be performed.	S3000L UoF Task
SubtaskSourceDocument	Class	relationship	SubtaskSourceDocument is a <<relationship>> that identifies an external Document where the Subtask is originally defined and more details are given.	S3000L UoF Task
SubtaskTarget	Class	relationship	SubtaskTarget is a <<relationship>> that identifies the item on which the Subtask is to be performed.	S3000L UoF Task
SubtaskTargetItem	Interface	select	SubtaskTargetItem is a <<select>> interface that identifies items on which a Subtask can be performed.	S3000L UoF Task
SubtaskTimeline	Class	relationship	SubtaskTimeline is a <<relationship>> that identifies that there is a time dependency between two Subtasks within the same Task.	S3000L UoF Task
SubtaskWarningCautionNote	Class	relationship	SubtaskWarningCautionNote is a <<relationship>> that identifies a WarningCautionNote that is associated with a given Subtask.	S3000L UoF Task

Class name	Type	Stereotype	Definition	UoF
SupportingTask	Class	class	SupportingTask is a Task that does not meet a TaskRequirement, but identifies a set of work steps which will be carried out as part of multiple Tasks. **Note** A SupportingTask will only be used in the context of SubtaskByTaskReference. **Note** The objective for a SupportingTask is to enable reuse of a sequence of work steps, needed by a set of Tasks. **Example** – Open hatch – Jack vehicle	S3000L UoF Task
SupportingTaskUsage	Class	relationship	SupportingTaskUsage is a TaskUsage that expands the definition of an embedded reusable Task in the context of a given support solution. **Note** A SupportingTask has no special characterizations apart from a TaskFrequency since it will never be performed on its own.	S3000L UoF Task Usage
Task	Class	class	Task is a <<class>> that represents the specification of work to be done or undertaken.	S3000L UoF Task
TaskAnalysisItem	Interface	extend	TaskAnalysisItem is an <<extend>> interface that provides its associated data model to those classes that implement it.	S3000L UoF Task Usage
TaskDocumentResource	Class	class	TaskDocumentResource is a TaskResource that identifies a Document used as a resource. **Example** – A form that must be filled out before, during or after the specified amount of work is carried out	S3000L UoF Task Resource
TaskFrequency	Class	attributeGroup	TaskFrequency is an <<attributeGroup>> that specifies the rate of occurrence of a Task in its defined usage.	S3000L UoF Task Usage

Class name	Type	Stereotype	Definition	UoF
TaskInfrastructureResource	Class	class	TaskInfrastructureResource is a TaskResource that defines foundational systems and services required as a resource.	S3000L UoF Task Resource
TaskJustification	Class	relationship	TaskJustification is a <<relationship>> that identifies a TaskRequirement that defines the need for the existence of a task.	S3000L UoF Task
TaskLocationItem	Interface	select	TaskLocationItem is a <<select>> interface that identifies items where Tasks can be performed.	S3000L UoF Task Usage
TaskMaterialResource	Class	class	TaskMaterialResource is a TaskResource that identifies parts which are required as a resource.	S3000L UoF Task Resource
TaskPersonnelResource	Class	class	TaskPersonnelResource is a TaskResource that specifies the man power required as a resource.	S3000L UoF Task Resource
TaskPersonnelResourceCompetence	Class	relationship	TaskPersonnelResourceCompetence is a <<relationship>> that identifies the proficiency required for TaskPersonnelResource.	S3000L UoF Task Resource
TaskRequirement	Class	class	TaskRequirement is a <<class>> that represents the need for a procedure to be developed and documented. **Note** A task requirement can have more than one task being developed for different usage scenarios. **Note** Task requirements are identified and documented prior to any detailed task analysis. **Note** Examples of support analysis activities which result in a set of documented task requirements are: Preventive Maintenance Analysis (PMA) (refer to S4000P) and special event analysis.	S3000L UoF Task Requirement
TaskRequirementAnalysisItem	Interface	extend	TaskRequirementAnalysisItem is an <<extend>> interface that provides its associated data model to those classes that implement it.	S3000L UoF Task Requirement

Class name	Type	Stereotype	Definition	UoF
TaskRequirementJustification	Class	relationship	TaskRequirementJustification is a <<relationship>> that identifies a source which defines the need for a Task.	S3000L UoF Task Requirement
TaskRequirementJustificationItem	Interface	select	TaskRequirementJustificationItem is a <<select>> interface that identifies items which can be selected as being the source that justifies the TaskRequirement.	S3000L UoF Task Requirement
TaskRequirementRevision	Class	class	TaskRequirementRevision is a <<class>> representing an iteration applied to a TaskRequirement.	S3000L UoF Task Requirement
TaskRequirements	Class	exchange	TaskRequirements is a wrapper element that contains all instances of TaskRequirement that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content
TaskResource	Class	class	TaskResource is a <<class>> that identifies means that have to be available to perform a specified amount of work.	S3000L UoF Task Resource
TaskResourceDefinitionItem	Interface	select	TaskResourceDefinitionItem is a <<select>> interface that identifies items which can be used as either infrastructure or material resources.	S3000L UoF Task Resource
TaskResourceItem	Interface	extend	TaskResourceItem is an <<extend>> interface that provides its associated data model to those classes that implement it.	S3000L UoF Task Resource
TaskResourceRelationship	Class	relationship	TaskResourceRelationship is a <<relationship>> where one TaskResource relates to another TaskResource.	S3000L UoF Task Resource
TaskRevision	Class	class	TaskRevision is a <<class>> representing an iteration applied to a Task.	S3000L UoF Task
TaskRevisionWarningCautionNote	Class	relationship	TaskRevisionWarningCautionNote is a <<relationship>> that identifies a WarningCautionNote that is associated with a given Task.	S3000L UoF Task
Tasks	Class	exchange	Tasks is a wrapper element that contains all instances of Task that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content
TaskUsage	Class	relationship	TaskUsage is a <<relationship>> that identifies a Task required for the TaskAnalysisItem.	S3000L UoF Task Usage

Class name	Type	Stereotype	Definition	UoF
TechnologyBehaviorRating	Class	attributeGroup	TechnologyBehaviorRating is an <<attributeGroup>> that collects attributes which categorizes the AnalysisCandidateItem based on knowledge on technology sensitivity and behavior perspective.	S3000L UoF Damage Definition
ThresholdDefinition	Class	class	ThresholdDefinition is a <<class>> that represents the circumstance that is used as a trigger or threshold.	S3000L UoF Time Limit
TimeLimit	Class	class	TimeLimit is a <<class>> that represents the specification of circumstances under which the associated item is initiated. **Note** TimeLimit does not have the concept of an identification which means that if there is a change to a TimeLimit for a TimeLimitItem, then all instances of TimeLimit must be re-sent as part of a Message. **Note** Time is used in the sense of "time to do something" and must not be seen as only a period of time.	S3000L UoF Time Limit
TimeLimitEventItem	Interface	select	TimeLimitEventItem is a <<select>> interface that identifies which items can be used to define an EventThresholdDefinition.	S3000L UoF Time Limit
TimeLimitItem	Interface	extend	TimeLimitItem is an <<extend>> interface that provides its associated data model to those classes that implement it.	S3000L UoF Time Limit
Trade	Class	class	Trade is a <<class>> that represents a craft or profession which requires specific skills.	S3000L UoF Competence Definition
Trades	Class	exchange	Trades is a wrapper element that contains all instances of Trade that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content
UsableOnItem	Interface	extend	UsableOnItem is an <<extend>> interface that provides its associated data model to those classes that can have a limited effectivity with respect to its usage in one or many instances of ProductVariant.	S3000L UoF Product Design Configuration

Class name	Type	Stereotype	Definition	UoF
UsableOnProductVariant	Class	relationship	UsableOnProductVariant is a <<relationship>> that defines that a UsableOnItem, included in the Breakdown for the overall Product, is effective in the associated ProductVariant. **Note** UsableOnProductVariant is the equivalent of the Usable On Code in SAE STD-GEIA-0007.	S3000L UoF Product Design Configuration
WarningCautionNote	Class	class	WarningCautionNote is a <<class>> that defines advice concerning safety, legal and health aspects.	S3000L UoF Task
WarningCautionNotes	Class	exchange	WarningCautionNotes is a wrapper element that contains all instances of WarningCautionNote that are in scope for a data exchange.	S3000L UoF Logistics Support Analysis Message Content
ZoneElement	Class	class	ZoneElement is a BreakdownElement that represents a three-dimensional space related to a Product. **Note** A zone can also represent a work area such as a mechanical workshop onboard a ship.	S3000L UoF Zone Element
ZoneElementRevision	Class	class	ZoneElementRevision is a BreakdownElementRevision representing an iteration applied to a ZoneElement.	S3000L UoF Zone Element

3 Class attribute list

The full list of S3000L attributes is provided in Table 3.

Table 3 is organized alphabetically by class attribute name and contains:

- Attribute name
- Attribute data type (refer to SX004G for more details on data types used in S3000L)
- Attribute definition which contains a textual definition
- Class name identifies the class in the S3000L data model where the attribute is used (refer to Chap 19)
- Unit of Functionality (UoF) identifies the section in Chap 19 where the class is defined

Table 3 List of class attributes

Attribute Name	Type	Definition	Class Name	UoF
additionalAddressInformation	DescriptorType	additionalAddressInformation is a description that provides additional information to further locate an address. **Example** – Building 7 in campus – First floor, apartment 7 – Suite 204	StreetAddress	S3000L UoF Location
additionalTrainingNeedDescription	DescriptorType	additionalTrainingNeedDescription is a description that gives more information on additional learning required. **Example** – Behaviors – Experience – Knowledge – Skills – Values	AdditionalTrainingNeed	S3000L UoF Task Resource
aggregatedElementType	ClassificationType	aggregatedElementType is a classification that identifies further specialization for an AggregatedElement.	AggregatedElement	S3000L UoF Aggregated Element
allowedProductConfigurationIdentifier	IdentifierType	allowedProductConfigurationIdentifier is an identifier that establishes a unique designator for an AllowedProductConfigurationByConfigurationIdentifier and to differentiate it from other instances of AllowedProductConfigurationByConfigurationIdentifier.	AllowedProductConfigurationByConfigurationIdentifier	S3000L UoF Product Design Configuration
altitude	umlString	altitude is a string of characters that represents the height above or below a fixed reference point. **Example** – 34m above sea level	GlobalPosition	S3000L UoF Location

Attribute Name	Type	Definition	Class Name	UoF
analysisActivityDecision	ClassificationType	analysisActivityDecision is a classification that identifies if the AnalysisActivity is to be performed on the AnalysisCandidateItem.	AnalysisActivityRevision	S3000L UoF LSA Candidate
analysisActivityDecisionRationale	DescriptorType	analysisActivityDecisionRationale is a description that gives more information on the reason for the selection / non-selection of the AnalysisActivity.	AnalysisActivityRevision	S3000L UoF LSA Candidate
analysisActivityRevisionDate	DateType	analysisActivityRevisionDate is a date that specifies when an AnalysisActivityRevision was defined.	AnalysisActivityRevision	S3000L UoF LSA Candidate
analysisActivityRevisionIdentifier	IdentifierType	analysisActivityRevisionIdentifier is an identifier that establishes a unique designator for an AnalysisActivityRevision and to differentiate it from other instances of AnalysisActivityRevision.	AnalysisActivityRevision	S3000L UoF LSA Candidate
analysisActivityRevisionRationale	DescriptorType	analysisActivityRevisionRationale is a description that gives more information on the justification for revising the defined AnalysisActivity.	AnalysisActivityRevision	S3000L UoF LSA Candidate
analysisActivityRevisionStatus	StateType	analysisActivityRevisionStatus is a state that identifies the maturity of an AnalysisActivityRevision.	AnalysisActivityRevision	S3000L UoF LSA Candidate
analysisActivityStatusDescription	DescriptorType	analysisActivityStatusDescription is a description that gives further information on the progression of the AnalysisActivity.	AnalysisActivityRevision	S3000L UoF LSA Candidate
analysisActivityType	ClassificationType	analysisActivityType is a classification that identifies further specialization of AnalysisActivity.	AnalysisActivity	S3000L UoF LSA Candidate
analysisCandidateItemSelectionIndicator	ClassificationType	analysisCandidateItemSelectionIndicator is a classification that specifies the extent to which analysis activities will be carried out on the associated AnalysisCandidateItem.	AnalysisCandidateItemSelectionData	S3000L UoF LSA Candidate

Attribute Name	Type	Definition	Class Name	UoF
analysisCandidateItemSelectionRationale	DescriptorType	analysisCandidateItemSelectionRationale is a description that gives more information on the reason for the candidate selection.	AnalysisCandidateItemSelectionData	S3000L UoF LSA Candidate
analysisCandidateMaintenanceConcept	DescriptorType	analysisCandidateMaintenanceConcept is a statement of maintenance considerations, constraints and strategy for the operational support that governs the maintenance levels and type of maintenance activities to be carried out on the AnalysisCandidateItem.	AnalysisCandidateItemSelectionData	S3000L UoF LSA Candidate
analysisCandidateMaintenanceSolution	DescriptorType	analysisCandidateMaintenanceSolution is a statement of maintenance activities and maintenance levels that have been decided for the AnalysisCandidateItem.	AnalysisCandidateItemSelectionData	S3000L UoF LSA Candidate
applicabilityStatementDateRange	DateRange	applicabilityStatementDateRange is a date range that defines the date interval for when the applicability evaluation can result in a TRUE result. **Note** If outside that date range, the ApplicabilityStatement always results in a FALSE statement.	ApplicabilityStatement	S3000L UoF Applicability Statement
applicabilityStatementDescription	DescriptorType	applicabilityStatementDescription is a description that provides a human readable expression of the defined rule.	ApplicabilityStatement	S3000L UoF Applicability Statement
applicabilityStatementIdentifier	IdentifierType	applicabilityStatementIdentifier is an identifier that establishes a unique designator for an ApplicabilityStatement and to differentiate it from other instances of ApplicabilityStatement.	ApplicabilityStatement	S3000L UoF Applicability Statement
applicableSerialNumberRange	SerialNumberRange	applicableSerialNumberRange is a serial number range that identifies a limited effectivity with respect to a given interval of serialized items.	UsableOnProductVariant	S3000L UoF Product Design Configuration
			EffectiveOnProductConfiguration	S3000L UoF Product Design Configuration

Attribute Name	Type	Definition	Class Name	UoF
			EvaluationByAssertionOfSerializedItems	S3000L UoF Applicability Statement
assembly	umlString	assembly is a string of characters that represents the unit or assembly attribute of the data module code.	S1000DDataModule	S3000L UoF Document
authorityToOperateIdentifier	IdentifierType	authorityToOperateIdentifier is an identifier that establishes a unique designator for an AuthorityToOperate and to differentiate it from other instances of AuthorityToOperate.	AuthorityToOperate	S3000L UoF Product Design Configuration
breakdownElementChildSequenceNumber	umlString	breakdownElementChildSequenceNumber is a string of characters that controls the order for the included child element. **Note** The sequence number can be used to control how child elements are presented in eg, a list.	BreakdownElementStructure	S3000L UoF Breakdown Structure
breakdownElementDescription	DescriptorType	breakdownElementDescription is a description that gives more information on the BreakdownElement.	BreakdownElementRevision	S3000L UoF Breakdown Structure
breakdownElementEssentiality	ClassificationType	breakdownElementEssentiality is a classification that identifies the operational importance of the BreakdownElement at the Product level. **Note** Based on the criticality as defined during the FMECA.	BreakdownElement	S3000L UoF Breakdown Structure

Attribute Name	Type	Definition	Class Name	UoF
breakdownElementIdentifier	IdentifierType	breakdownElementIdentifier is an identifier that establishes a unique designator for a BreakdownElement and to differentiate it from other instances of BreakdownElement. **Note** Can be used to establish a hierarchical structure of the technical system. **Example** – The combination of Logistics Support Analysis Control Number (LCN) and Alternate Logistics Support Analysis Control Number (ALC) within SAE GEIA-STD-0007 – The Standard Numbering System (SNS) defined by S1000D	BreakdownElement	S3000L UoF Breakdown Structure
breakdownElementName	NameType	breakdownElementName is a name by which the BreakdownElement is known and can be easily referenced.	BreakdownElement	S3000L UoF Breakdown Structure
breakdownElementRevisionDate	DateType	breakdownElementRevisionDate is a date that specifies when the BreakdownElement was revised.	BreakdownElementRevision	S3000L UoF Breakdown Structure
breakdownElementRevisionIdentifier	IdentifierType	breakdownElementRevisionIdentifier is an identifier that establishes a unique designator for a BreakdownElementRevision and to differentiate it from other instances of BreakdownElementRevision.	BreakdownElementRevision	S3000L UoF Breakdown Structure
breakdownElementRevisionRationale	DescriptorType	breakdownElementRevisionRationale is a description that gives more information on the justification for revising the BreakdownElement.	BreakdownElementRevision	S3000L UoF Breakdown Structure
breakdownElementRevisionRelationshipType	ClassificationType	breakdownElementRevisionRelationshipType is a classification that identifies the meaning of the established relationship. **Note** The related breakdown elements do not need to be used in the same breakdown ie, it can be used to establish the relationship between a breakdown element in a functional breakdown and a breakdown element in a physical breakdown.	BreakdownElementRevisionRelationship	S3000L UoF Breakdown Structure

Attribute Name	Type	Definition	Class Name	UoF
breakdownElementRevisionStatus	StateType	breakdownElementRevisionStatus is a state that identifies the maturity of a BreakdownElementRevision.	BreakdownElementRevision	S3000L UoF Breakdown Structure
breakdownElementUsageIdentifier	IdentifierType	breakdownElementUsageIdentifier is an identifier that establishes a unique designator for a BreakdownElementUsageInBreakdown and to differentiate it from other instances of BreakdownElementUsageInBreakdown.	BreakdownElementUsageInBreakdown	S3000L UoF Breakdown Structure
breakdownElementUsageQuantity	PropertyType	breakdownElementUsageQuantity is a property that specifies the amount of the BreakdownElement used in its parent BreakdownElement. **Note** If no value is given, it must be interpreted as value "1" with a unit of "each". For as required amounts, the text property is used with "As Required" or other text as appropriate.	BreakdownElementUsageInBreakdown	S3000L UoF Breakdown Structure
breakdownElementUsageRelationshipType	ClassificationType	breakdownElementUsageRelationshipType is a classification that identifies the meaning of the established relationship.	BreakdownElementUsageRelationship	S3000L UoF Breakdown Structure
breakdownRevisionDate	DateType	breakdownRevisionDate is a date that specifies when the Breakdown was revised.	BreakdownRevision	S3000L UoF Breakdown Structure
breakdownRevisionIdentifier	IdentifierType	breakdownRevisionIdentifier is an identifier that establishes a unique designator for a BreakdownRevision and to differentiate it from other instances of BreakdownRevision.	BreakdownRevision	S3000L UoF Breakdown Structure
breakdownRevisionRationale	DescriptorType	breakdownRevisionRationale is a description that gives more information on the justification for revising the Breakdown.	BreakdownRevision	S3000L UoF Breakdown Structure
breakdownRevisionRelationshipType	ClassificationType	breakdownRevisionRelationshipType is a classification that identifies the meaning of the established relationship.	BreakdownRevisionRelationship	S3000L UoF Breakdown Structure
breakdownRevisionStatus	StateType	breakdownRevisionStatus is a state that identifies the maturity of a BreakdownRevision.	BreakdownRevision	S3000L UoF Breakdown Structure

Attribute Name	Type	Definition	Class Name	UoF
breakdownType	ClassificationType	breakdownType is a classification that identifies the perspective from which the Breakdown is defined.	Breakdown	S3000L UoF Breakdown Structure
changeAuthorizationIdentifier	IdentifierType	changeAuthorizationIdentifier is an identifier that establishes a unique designator for a ChangeAuthorization and to differentiate it from other instances of ChangeAuthorization.	ChangeAuthorization	S3000L UoF Change Information
changeNotificationDescription	DescriptorType	changeNotificationDescription is a description providing a summary of affects made to the related item due to a ChangeAuthorization.	ChangeNotification	S3000L UoF Change Information
changeNotificationType	ClassificationType	changeNotificationType is a classification that identifies a change effect as belonging to a group of change effects sharing a particular characteristic or set of characteristics.	ChangeNotification	S3000L UoF Change Information
changeRequestDescription	DescriptorType	changeRequestDescription is a description, providing detailed explanation of the desired change.	ChangeRequest	S3000L UoF Design Change Request
changeRequestIdentifier	IdentifierType	changeRequestIdentifier is an identifier that establishes a unique designator for a ChangeRequest and to differentiate it from other instances of ChangeRequest.	ChangeRequest	S3000L UoF Design Change Request
changeRequestIntendedEffect	DescriptorType	changeRequestIntendedEffect is a description, providing detailed explanation of the expected effect of the change.	ChangeRequest	S3000L UoF Design Change Request
changeRequestStatus	StateType	changeRequestStatus is a state, that documents the progress of the ChangeRequest within its lifecycle.	ChangeRequest	S3000L UoF Design Change Request
circuitBreakerIdentifier	IdentifierType	circuitBreakerIdentifier is an identifier that establishes a unique designator for a CircuitBreaker and to differentiate it from other instances of CircuitBreaker.	CircuitBreaker	S3000L UoF Circuit Breaker
circuitBreakerLocationDescription	DescriptorType	circuitBreakerLocationDescription is a description that provides further details on where the CircuitBreaker is located on the referenced CircuitBreakerLocationItem.	CircuitBreakerLocation	S3000L UoF Circuit Breaker

Attribute Name	Type	Definition	Class Name	UoF
circuitBreakerName	NameType	circuitBreakerName is a name by which the CircuitBreaker is known and can be easily referenced.	CircuitBreaker	S3000L UoF Circuit Breaker
circuitBreakerSettingIdentifier	IdentifierType	circuitBreakerSettingIdentifier is an identifier that establishes a unique designator for a defined CircuitBreakerSetting, and to differentiate it from other instances of CircuitBreakerSetting.	CircuitBreakerSetting	S3000L UoF Task
circuitBreakerSettingsIdentifier	IdentifierType	circuitBreakerSettingsIdentifier is an identifier that establishes a unique designator for a defined set of circuit breaker settings and to differentiate it from other instances of circuit breaker settings.	CircuitBreakerSettings	S3000L UoF Task
circuitBreakerSettingsOrdered	umlBoolean	circuitBreakerSettingsOrdered is a boolean that defines if the individual circuit breaker setting must be performed in the specified order. **Note** A TRUE value specifies that the circuit breaker settings must be accomplished in the defined order. A FALSE value specifies that the circuit breaker settings can be accomplished in any order.	CircuitBreakerSettings	S3000L UoF Task
circuitBreakerState	StateType	circuitBreakerState is a state that identifies the position that a given CircuitBreaker must be in after the accomplishment of a defined CircuitBreakerSetting.	CircuitBreakerSetting	S3000L UoF Task
circuitBreakerType	ClassificationType	circuitBreakerType is a classification that defines the technical principle for the CircuitBreaker.	CircuitBreaker	S3000L UoF Circuit Breaker
cityName	NameType	cityName is a name by which an incorporated municipal unit is known and can be easily referenced.	StreetAddress	S3000L UoF Location
competencyDefinitionDescription	DescriptorType	competencyDefinitionDescription is a description that provides information on the knowledge, skills and attitudes that a person is expected to have by having acquired a certain competency.	CompetencyDefinition	S3000L UoF Competence Definition

Attribute Name	Type	Definition	Class Name	UoF
competencyDefinitionIdentifier	IdentifierType	competencyDefinitionIdentifier is an identifier that establishes a unique designator for a CompetencyDefinition and to differentiate it from other instances of CompetencyDefinition.	CompetencyDefinition	S3000L UoF Competence Definition
competencyDefinitionName	NameType	competencyDefinitionName is a name by which the CompetencyDefinition is known and can be easily referenced.	CompetencyDefinition	S3000L UoF Competence Definition
competencyDefinitionType	ClassificationType	competencyDefinitionType is a classification that identifies further specialization of CompetencyDefinition.	CompetencyDefinition	S3000L UoF Competence Definition
conditionInstanceDescription	DescriptorType	conditionInstanceDescription is a description that gives more information on the meaning of the ConditionInstance.	ConditionInstance	S3000L UoF Applicability Statement
conditionInstanceIdentifier	IdentifierType	conditionInstanceIdentifier is an identifier that establishes a unique designator for a ConditionInstance and to differentiate it from other instances of ConditionInstance.	ConditionInstance	S3000L UoF Applicability Statement
conditionInstanceName	NameType	conditionInstanceName is a name by which the ConditionInstance is known and can be easily referenced.	ConditionInstance	S3000L UoF Applicability Statement
conditionTypeAssertMemberAssertValue	PropertyType	conditionTypeAssertMemberAssertValue is a numerical property that specifies values which can be used to further characterize the ConditionTypeAssertMember.	ConditionTypeAssertMember	S3000L UoF Applicability Statement
conditionTypeAssertMemberAssertValueComparisonOperator	ClassificationType	conditionTypeAssertMemberAssertValueComparisonOperator is a classification that identifies a mathematical operation to be applied when testing a value against a defined conditionTypeAssertMemberAssertValue. **Example** – Greater than – Less than	ConditionTypeAssertMember	S3000L UoF Applicability Statement
conditionTypeAssertMemberDescription	DescriptorType	conditionTypeAssertMemberDescription is a description that gives more information on the meaning of the ConditionTypeAssertMember.	ConditionTypeAssertMember	S3000L UoF Applicability Statement

Attribute Name	Type	Definition	Class Name	UoF
conditionTypeAssertMemberName	NameType	conditionTypeAssertMemberName is a name that identifies a ConditionTypeAssertMember.	ConditionTypeAssertMember	S3000L UoF Applicability Statement
conditionTypeDescription	DescriptorType	conditionTypeDescription is a description that gives more information on the meaning of the ConditionType.	ConditionType	S3000L UoF Applicability Statement
conditionTypeName	NameType	conditionTypeName is a name by which the ConditionType is known and can be easily referenced. **Example** – Ashore or afloat – Maintenance environment – Operational environment – Service bulletin	ConditionType	S3000L UoF Applicability Statement
containedSubstanceJustification	DescriptorType	containedSubstanceJustification is a description that gives more information on the properties of the included substance required for the function of the HardwarePartAsDesigned.	ContainedSubstance	S3000L UoF Part Definition
contractIdentifier	IdentifierType	contractIdentifier is an identifer that establishes a unique designator for a Contract and to differentiate it from other instances of Contract.	Contract	S3000L UoF Product and Project
contractItemDetailsBlockOfSerializedItems	SerialNumberRange	contractItemDetailsBlockOfSerializedItems is a serial number range that identifies an interval of serialized items as known by the customer.	ContractItemDetails	S3000L UoF Product and Project
contractItemDetailsContractQuantity	PropertyType	contractItemDetailsContractQuantity is a property that identifies the number of contract items that are included in the Contract.	ContractItemDetails	S3000L UoF Product and Project
contractName	NameType	contractName is a name by which the Contract is known and can be easily referenced.	Contract	S3000L UoF Product and Project

Attribute Name	Type	Definition	Class Name	UoF
contractPartyRole	ClassificationType	contractPartyRole is a classification that defines the purpose of the association between a ContractParty and the Contract. **Example** – Contractor – Customer – Escrow holder – Subcontractor – Supplier	ContractParty	S3000L UoF Product and Project
contractRelationshipType	ClassificationType	contractRelationshipType is a classification that identifies the meaning of the established relationship. **Example** – Associated – Extends – Replaces – Subcontract	ContractRelationship	S3000L UoF Product and Project
correctionFactorJustification	DescriptorType	correctionFactorJustification is a description that gives more information on the reason for the CorrectionFactor and for the correction value to be applied.	CorrectionFactor	S3000L UoF Performance Parameter
correctionFactorValue	umlReal	correctionFactorValue is a <<umlReal>> that specifies the amount by which the associated value is to be adjusted in the defined context.	CorrectionFactor	S3000L UoF Performance Parameter
countryCode	ClassificationType	countryCode is a string of characters used to uniquely identify a Country and to differentiate it from other instances of Country. **Note** It is advised to use ISO 3166-1 alpha-2.	Country	S3000L UoF Location
damageDefinitionDescription	DescriptorType	damageDefinitionDescription is a description that gives more information on the DamageDefinition.	DamageDefinition	S3000L UoF Damage Definition

Attribute Name	Type	Definition	Class Name	UoF
damageDefinitionFamily	ClassificationType	damageDefinitionFamily is a classification that identifies a group of damages which share the damage characteristics and often lead to the same, or similar, corrective action. **Example** – Crack – Dent – Scratch	DamageDefinition	S3000L UoF Damage Definition
damageDefinitionName	NameType	damageDefinitionName is a name by which the DamageDefinition is known and can be easily referenced.	DamageDefinition	S3000L UoF Damage Definition
damageImpactRatio	PropertyType	damageImpactRatio identifies the fraction of individual occurrences of DamageDefinition that will result in the associated FailureMode in relation to the entire population of occurences of the DamageDefinition identified for the AnalysisCandidateItem.	DamageImpact	S3000L UoF Damage Definition
dataModuleIssueInWorkNumber	umlString	dataModuleIssueInWorkNumber is a string of characters used for monitoring and control of intermediate drafts of S1000DDataModuleIssue. **Note** A dataModuleIssueInWorkNumber must be created in accordance with the rules defined in S1000D.	S1000DDataModuleIssue	S3000L UoF Document
dataModuleIssueLanguage	ClassificationType	dataModuleIssueLanguage is a classification that identifies the language used to produce the content of the S1000DDataModuleIssue. **Note** A dataModuleIssueLanguage must be created in accordance with the rules defined in S1000D.	S1000DDataModuleIssue	S3000L UoF Document

Attribute Name	Type	Definition	Class Name	UoF
dataModuleIssueLanguageCountry	ClassificationType	dataModuleIssueLanguageCountry is a classification that identifies the country where the language, identified by dataModuleIssueLanguage, is spoken **Note** A dataModuleIssueLanguageCountry must be created in accordance with the rules defined in S1000D.	S1000DDataModuleIssue	S3000L UoF Document
dataModuleIssueNumber	umlString	dataModuleIssueNumber is a string of characters used to identify the release number of the S1000DDataModuleIssue **Note** A dataModuleIssueNumber must be created in accordance with the rules defined in S1000D.	S1000DDataModuleIssue	S3000L UoF Document
decisionTreeTemplateActionDefinitionDescription	DescriptorType	decisionTreeTemplateActionDefinitionDescription is a description that provides further information on the action to be taken.	DecisionTreeTemplateActionDefinition	S3000L UoF Decision Tree Template Definition
decisionTreeTemplateActionDefinitionIdentifier	IdentifierType	decisionTreeTemplateActionDefinitionIdentifier is an identifier that establishes a unique designator for a DecisionTreeTemplateActionDefinition and to differentiate it from other instances of DecisionTreeTemplateActionDefinition.	DecisionTreeTemplateActionDefinition	S3000L UoF Decision Tree Template Definition
decisionTreeTemplateActionDefinitionName	NameType	decisionTreeTemplateActionDefinitionName is a name that defines the core action and/or measure to be taken.	DecisionTreeTemplateActionDefinition	S3000L UoF Decision Tree Template Definition

Attribute Name	Type	Definition	Class Name	UoF
decisionTreeTemplateAnalysisDomain	NameType	decisionTreeTemplateAnalysisDomain is a name that further specifies the domain for which the DecisionTreeTemplate is defined. **Example** – Gear boxes – In Service Support Optimization – In Service Training Optimization – Logistics Support Analysis – Obsolescence	DecisionTreeTemplateRevision	S3000L UoF Decision Tree Template Definition
decisionTreeTemplateDescription	DescriptorType	decisionTreeTemplateDescription is a description that provides more information on the purpose and scope of the defined analysis.	DecisionTreeTemplateRevision	S3000L UoF Decision Tree Template Definition
decisionTreeTemplateIdentifier	IdentifierType	decisionTreeTemplateIdentifier is an identifier that establishes a unique designator for a DecisionTreeTemplate and to differentiate it from other instances of DecisionTreeTemplate.	DecisionTreeTemplate	S3000L UoF Decision Tree Template Definition
decisionTreeTemplateName	NameType	decisionTreeTemplateName is a name by which the DecisionTreeTemplate is known and can be easily referenced.	DecisionTreeTemplate	S3000L UoF Decision Tree Template Definition
decisionTreeTemplateQuestionDefinitionDescription	DescriptorType	decisionTreeTemplateQuestionDefinitionDescription is a description that provides further information and clarification about the question to be answered.	DecisionTreeTemplateQuestionDefinition	S3000L UoF Decision Tree Template Definition
decisionTreeTemplateQuestionDefinitionIdentifier	IdentifierType	decisionTreeTemplateQuestionDefinitionIdentifier is an identifier that establishes a unique designator for a DecisionTreeTemplateQuestionDefinition and to differentiate it from other instances of DecisionTreeTemplateQuestionDefinition.	DecisionTreeTemplateQuestionDefinition	S3000L UoF Decision Tree Template Definition
decisionTreeTemplateQuestionDefinitionName	NameType	decisionTreeTemplateQuestionDefinitionName is a name that defines the core question to be answered.	DecisionTreeTemplateQuestionDefinition	S3000L UoF Decision Tree Template Definition

Attribute Name	Type	Definition	Class Name	UoF
decisionTreeTemplateRevisionDate	DateType	decisionTreeTemplateRevisionDate is a date that specifies when a DecisionTreeTemplateRevision was defined.	DecisionTreeTemplateRevision	S3000L UoF Decision Tree Template Definition
decisionTreeTemplateRevisionIdentifier	IdentifierType	decisionTreeTemplateRevisionIdentifier is an identifier that establishes a unique designator for a DecisionTreeTemplateRevision and to differentiate it from other instances of DecisionTreeTemplateRevision.	DecisionTreeTemplateRevision	S3000L UoF Decision Tree Template Definition
decisionTreeTemplateRevisionRationale	DescriptorType	decisionTreeTemplateRevisionRationale is a description that gives more information on the justification for revising the defined DecisionTreeTemplate.	DecisionTreeTemplateRevision	S3000L UoF Decision Tree Template Definition
decisionTreeTemplateRevisionStatus	StateType	decisionTreeTemplateRevisionStatus is a state that identifies the maturity of a DecisionTreeTemplateRevision.	DecisionTreeTemplateRevision	S3000L UoF Decision Tree Template Definition
detectionMeansAlarmDescription	DescriptorType	detectionMeansAlarmDescription is a description that gives more information on the DetectionMeansAlarm.	DetectionMeansAlarm	S3000L UoF Failure Mode Symptom
detectionMeansAlarmFalseAlarmRate	PropertyType	detectionMeansAlarmFalseAlarmRate identifies the portion of all alarms that indicate a malfunction that cannot be verified by personnel performing follow-on tests.	DetectionMeansAlarm	S3000L UoF Failure Mode Symptom
detectionMeansAlarmIdentifier	IdentifierType	detectionMeansAlarmIdentifier is an identifier that establishes a unique designator for a DetectionMeansAlarm and to differentiate it from other instances of DetectionMeansAlarm.	DetectionMeansAlarm	S3000L UoF Failure Mode Symptom
detectionMeansAlarmPresentation	DescriptorType	detectionMeansAlarmPresentation is a description that gives more information on how the alarm is presented.	DetectionMeansAlarm	S3000L UoF Failure Mode Symptom
detectionMeansAlarmType	ClassificationType	detectionMeansAlarmType is a classification that defines the type of test that generates the symptom.	DetectionMeansAlarm	S3000L UoF Failure Mode Symptom

Attribute Name	Type	Definition	Class Name	UoF
digitalFileContentClass	ClassificationType	digitalFileContentClass is a classification that determine the meaning of the information within the DigitalFile. **Example** – Audio – Courseware – Drawing – Movie – Problem report – Publication	DigitalFile	S3000L UoF Digital File
digitalFileContentDescription	DescriptorType	digitalFileContentDescription is a phrase that gives more details about the information contained in the DigitalFile.	DigitalFile	S3000L UoF Digital File
digitalFileLocator	IdentifierType	digitalFileLocator is an identifier that establishes a unique designator for a DigitalFile used to locate and identify a DigitalFile and to differentiate it from other instances of DigitalFile.	DigitalFile	S3000L UoF Digital File
digitalFileReferenceJustification	DescriptorType	digitalFileReferenceJustification is a description that provides more on information on the reason why the DigitalFileReferencedItem is referenced. **Example** – Crack discovered on BreakdownElement 'ABC-123'	DigitalFileReference	S3000L UoF Digital File
digitalFileRepresentation	umlString	digitalFileRepresentation is a string of characters representing the content of the DigitalFile. **Example** A uuencoded ASCII umlString representing a binary source file.	DigitalFile	S3000L UoF Digital File
digitalFileType	ClassificationType	digitalFileType is a classification that specifies the format of the data within the DigitalFile. **Note** Typically, the file name extension in Microsoft Windows.	DigitalFile	S3000L UoF Digital File

Attribute Name	Type	Definition	Class Name	UoF
disassemblyCode	umlString	disassemblyCode is a string of characters that represents the disassembly code attribute of the data module code. **Note** A disassemblyCode must be created in accordance with the rules defined in S1000D.	S1000DDataModule	S3000L UoF Document
disassemblyCodeVariant	umlString	disassemblyCodeVariant is a string of characters that represents the disassembly code variant attribute of the data module code. **Note** A disassemblyCodeVariant must be created in accordance with the rules defined in S1000D.	S1000DDataModule	S3000L UoF Document
documentIdentifier	IdentifierType	documentIdentifier is an identifier that establishes a unique designator for a Document and to differentiate it from other instances of Document.	Document	S3000L UoF Document
documentIssueDate	DateType	documentIssueDate is a date that defines when a specific issue of a document was released.	DocumentIssue	S3000L UoF Document
documentIssueIdentifier	IdentifierType	documentIssueIdentifier is an identifier that establishes a unique designator for a DocumentIssue and to differentiate it from other instances of DocumentIssue.	DocumentIssue	S3000L UoF Document
documentTitle	NameType	documentTitle is a name by which the Document is known and can be easily referenced.	Document	S3000L UoF Document
documentType	ClassificationType	documentType is a classification that identifies the category of the Document.	Document	S3000L UoF Document
environmentDefinitionCharacteristicDescription	DescriptorType	environmentDefinitionCharacteristicDescription is a description that gives more information on the EnvironmentDefinitionCharacteristic.	EnvironmentDefinitionCharacteristic	S3000L UoF Environment Definition

Attribute Name	Type	Definition	Class Name	UoF
environmentDefinitionCharacteristicName	NameType	environmentDefinitionCharacteristicName is a name by which the EnvironmentDefinitionCharacteristic is known and can be easily referenced.	EnvironmentDefinitionCharacteristic	S3000L UoF Environment Definition
environmentDefinitionCharacteristicValue	PropertyType	environmentDefinitionCharacteristicValue is a property that represents a measurable or observable characteristic for a circumstance, object, event, or condition that is typical to the EnvironmentDefinition.	EnvironmentDefinitionCharacteristic	S3000L UoF Environment Definition
environmentDefinitionCharacteristicValueComparisonOperator	ClassificationType	environmentDefinitionCharacteristicValueComparisonOperator is a classification that identifies the comparison operator which is to be used in order to qualify whether an actual environment is a member of the defined EnvironmentDefinition.	EnvironmentDefinitionCharacteristic	S3000L UoF Environment Definition
environmentDefinitionDescription	DescriptorType	environmentDefinitionDescription is a description that gives more information on the EnvironmentDefinition.	EnvironmentDefinitionRevision	S3000L UoF Environment Definition
environmentDefinitionIdentifier	IdentifierType	environmentDefinitionIdentifier is an identifier that establishes a unique designator for an EnvironmentDefinition and to differentiate it from other instances of EnvironmentDefinition.	EnvironmentDefinition	S3000L UoF Environment Definition
environmentDefinitionName	NameType	environmentDefinitionName is a name by which the EnvironmentDefinition is known and can be easily referenced.	EnvironmentDefinitionRevision	S3000L UoF Environment Definition
environmentDefinitionRelationshipType	ClassificationType	environmentDefinitionRelationshipType is a classification that identifies the meaning of the established relationship.	EnvironmentDefinitionRelationship	S3000L UoF Environment Definition
environmentDefinitionRevisionDate	DateType	environmentDefinitionRevisionDate is a date that specifies when the EnvironmentDefinition was revised.	EnvironmentDefinitionRevision	S3000L UoF Environment Definition
environmentDefinitionRevisionIdentifier	IdentifierType	environmentDefinitionRevisionIdentifier is an identifier that establishes a unique designator for an EnvironmentDefinitionRevision and to differentiate it from other instances of EnvironmentDefinitionRevision.	EnvironmentDefinitionRevision	S3000L UoF Environment Definition

Attribute Name	Type	Definition	Class Name	UoF
environmentDefinitionRevisionRationale	DescriptorType	environmentDefinitionRevisionRationale is a description that provides a justification for revising the EnvironmentDefinition.	EnvironmentDefinitionRevision	S3000L UoF Environment Definition
environmentDefinitionRevisionStatus	StateType	environmentDefinitionRevisionStatus is a state that identifies the maturity of an EnvironmentDefinitionRevision.	EnvironmentDefinitionRevision	S3000L UoF Environment Definition
evaluationByAssertionRole	ClassificationType	evaluationByAssertionRole is a classification that defines the context in which the EvaluationByAssertionOfClassInstance is being referenced.	EvaluationByAssertionOfClassInstance	S3000L UoF Applicability Statement
eventThresholdNumberOfEventOccurrences	umlInteger	eventThresholdNumberOfEventOccurrences is an integer that defines how many times an event must be repeated for the EventThresholdDefinition to be activated.	EventThresholdDefinition	S3000L UoF Time Limit
extensionCode	umlString	extensionCode is a string of characters used to identify the organization receiving the customized data module. **Note** An extensionCode must be created in accordance with the rules defined in S1000D.	S1000DDataModule	S3000L UoF Document
			S1000DPublicationModule	S3000L UoF Document
extensionProducer	umlString	extensionProducer is a string of characters used to identify the organization providing the customized data module. **Note** An extensionProducer must be created in accordance with the rules defined in S1000D.	S1000DDataModule	S3000L UoF Document
			S1000DPublicationModule	S3000L UoF Document
facilityDescription	DescriptorType	facilityDescription is a description that gives more information on capabilities provided by the Facility.	Facility	S3000L UoF Facility
facilityIdentifier	IdentifierType	facilityIdentifier is an identifier that establishes a unique designator for a Facility and to differentiate it from other instances of Facility.	Facility	S3000L UoF Facility

Attribute Name	Type	Definition	Class Name	UoF
facilityName	NameType	facilityName is a name by which the Facility is known and can be easily referenced.	Facility	S3000L UoF Facility
facilityRelationshipType	ClassificationType	facilityRelationshipType is a classification that identifies the meaning of the established relationship.	FacilityRelationship	S3000L UoF Facility
failureModeAnalysisDescription	DescriptorType	failureModeAnalysisDescription is a description that gives more information on the FailureModeAnalysis. **Example** – The failureModeAnalysisDescription can give further information on background and scope.	FailureModeAnalysisRevision	S3000L UoF Failure Mode
failureModeAnalysisRevisionDate	DateType	failureModeAnalysisRevisionDate is a date that specifies when the FailureModeAnalysis was revised.	FailureModeAnalysisRevision	S3000L UoF Failure Mode
failureModeAnalysisRevisionIdentifier	IdentifierType	failureModeAnalysisRevisionIdentifier is an identifier that establishes a unique designator for a FailureModeAnalysisRevision and to differentiate it from other instances of FailureModeAnalysisRevision.	FailureModeAnalysisRevision	S3000L UoF Failure Mode
failureModeAnalysisRevisionRationale	DescriptorType	failureModeAnalysisRevisionRationale is a description that gives more information on the justification for revising the FailureModeAnalysis.	FailureModeAnalysisRevision	S3000L UoF Failure Mode
failureModeAnalysisRevisionStatus	StateType	failureModeAnalysisRevisionStatus is a state that identifies the maturity of a FailureModeAnalysisRevision.	FailureModeAnalysisRevision	S3000L UoF Failure Mode
failureModeAnalysisType	ClassificationType	failureModeAnalysisType is a classification that identifies further specialization for FailureModeAnalysis.	FailureModeAnalysis	S3000L UoF Failure Mode
failureModeCauseDescription	DescriptorType	failureModeCauseDescription is a description that gives more information on the FailureModeCause.	FailureModeCause	S3000L UoF Failure Mode

Attribute Name	Type	Definition	Class Name	UoF
failureModeCauseIdentifier	IdentifierType	failureModeCauseIdentifier is an identifier that establishes a unique designator for a FailureModeCause and to differentiate it from other instances of FailureModeCause.	FailureModeCause	S3000L UoF Failure Mode
failureModeCauseItemRelationshipType	ClassificationType	failureModeCauseItemRelationshipType is a classification that identifies the meaning of the established relationship.	FailureModeCauseItemRelationship	S3000L UoF Failure Mode
failureModeCauseRatio	PropertyType	failureModeCauseRatio identifies the fraction of an individual FailureModeCause in relation to the entire population of FailureModeCauses identified for the FailureMode.	FailureModeCause	S3000L UoF Failure Mode
failureModeCriticality	ClassificationType	failureModeCriticality is a classification that identifies the most serious impact that the FailureMode will have on the referred item.	FailureModeEffect	S3000L UoF Failure Mode
failureModeDescription	DescriptorType	failureModeDescription is a description that gives more information on the FailureMode.	FailureMode	S3000L UoF Failure Mode
failureModeDetectionAbilityDescription	DescriptorType	failureModeDetectionAbilityDescription is a description that gives more information on the ability to detect the associated failure mode.	FailureModeSymptomsSignature	S3000L UoF Failure Mode Symptom
failureModeDetectionAbilityRatio	PropertyType	failureModeDetectionAbilityRatio is a property that identifies the fraction of failure mode occurrences that will be detected through the symptoms signature in relation to the entire population of occurrences for the associated FailureModeSymptomsSignatureItem.	FailureModeSymptomsSignature	S3000L UoF Failure Mode Symptom
failureModeEffectDescription	DescriptorType	failureModeEffectDescription is a description that gives more information on the FailureModeEffect.	FailureModeEffect	S3000L UoF Failure Mode
failureModeEffectLevel	ClassificationType	failureModeEffectLevel is classification that identifies the higher indenture level that will be affected by the FailureMode.	FailureModeEffect	S3000L UoF Failure Mode
failureModeEffectName	NameType	failureModeEffectName is a name by which the FailureModeEffect is known and can be easily referenced.	FailureModeEffect	S3000L UoF Failure Mode

Attribute Name	Type	Definition	Class Name	UoF
failureModeEffectSummaryDescription	DescriptorType	failureModeEffectSummaryDescription is a description that provides a summary of human observable failure mode effects that indicates that the associated failure mode has occurred.	FailureModeSymptomsSignature	S3000L UoF Failure Mode Symptom
failureModeIdentifier	IdentifierType	failureModeIdentifier is an identifier that establishes a unique designator for a FailureMode and to differentiate it from other instances of FailureMode.	FailureMode	S3000L UoF Failure Mode
failureModeIsolationAbilityDescription	DescriptorType	failureModeIsolationAbilityDescription is a description that gives more information on the ability to isolate the associated failure mode.	FailureModeIsolationCharacteristics	S3000L UoF Failure Mode Isolation
failureModeIsolationAbilityRating	ClassificationType	failureModeIsolationAbilityRating is a classification that defines the ability to isolate the associated failure mode(s).	FailureModeIsolationCharacteristics	S3000L UoF Failure Mode Isolation
failureModeIsolationRate	PropertyType	failureModeIsolationRate defines the fraction of detected failures that can be associated to a failure of an identified item or component without ambiguity.	FailureModeIsolationCharacteristics	S3000L UoF Failure Mode Isolation
failureModeName	NameType	failureModeName is a name by which the FailureMode is known and can be easily referenced.	FailureMode	S3000L UoF Failure Mode
failureModeRatio	PropertyType	failureModeRatio identifies the fraction of an individual FailureMode in relation to the entire population of FailureModes identified for the FailureModeAnalysisItem. **Note** If the failureModeRatio equals '0' then the failure mode is only driven by damage.	FailureMode	S3000L UoF Failure Mode
failureModeSymptomRatio	PropertyType	failureModeSymptomRatio is a property that specifies the fraction of failure mode occurrences that will be detected through the associated symptom in relation to the entire population of occurrences for the associated FaiureModeSymptomsSignatureItem.	FailureModeSymptom	S3000L UoF Failure Mode Symptom

Attribute Name	Type	Definition	Class Name	UoF
failureModeSymptomsDescription	DescriptorType	failureModeSymptomsDescription is a description that provides a human readable summary of the set of symptoms that together indicates that the associated failure mode have occurred.	FailureModeSymptomsSignature	S3000L UoF Failure Mode Symptom
failureModeSymptomsSignatureIdentifier	IdentifierType	failureModeSymptomsSignatureIdentifier is an identifier that establishes a unique designator for a FailureModeSymptomsSignature and to differentiate it from other instances of FailureModeSymptomsSignature.	FailureModeSymptomsSignature	S3000L UoF Failure Mode Symptom
followOnInServiceOptimizationAnalysisRationale	DescriptorType	followOnInServiceOptimizationAnalysisRationale is a description that gives more information on the reason for the additionally initiated InServiceOptimizationAnalysis.	FollowOnInServiceOptimizationAnalysis	S3000L UoF In Service Optimization Analysis
geographicalAreaDescription	DescriptorType	geographicalAreaDescription is a description that provides more information about the GeographicalArea.	GeographicalArea	S3000L UoF Location
geographicalAreaName	NameType	geographicalAreaName is a name by which the GeographicalArea is known and can be easily referenced. **Example** – Central alps – Dade county – Europe – Gobi desert – Tokyo – USA	GeographicalArea	S3000L UoF Location
geographicalAreaType	ClassificationType	geographicalAreaType is a classification that identifies the nature of the GeographicalArea.	GeographicalArea	S3000L UoF Location
geographicalCoordinateSystem	ClassificationType	geographicalCoordinateSystem is a classification that identifies the geographical coordinate system used to determine latitude and longitude.	GlobalPosition	S3000L UoF Location

Attribute Name	Type	Definition	Class Name	UoF
hardwareElementRepairability	ClassificationType	hardwareElementRepairability is a classification that indicates whether the HardwareElement part realization is expected to be reparable from a technical standpoint, independent of customer maintenance concepts.	HardwareElementRevision	S3000L UoF Hardware Element
hardwareElementRepairabilityStrategy	ClassificationType	hardwareElementRepairabilityStrategy is a classification that identifies the reparability strategy chosen for a specific customer and maintenance concept. **Note** Repairability strategy can include information on which maintenance level the repair is to be performed.	HardwareElementRevision	S3000L UoF Hardware Element
hardwareElementReplaceability	ClassificationType	hardwareElementReplaceability is a classification that identifies whether the HardwareElement part realization is expected to be replaceable from a technical standpoint, independent from customer maintenance concepts.	HardwareElementRevision	S3000L UoF Hardware Element
hardwareElementReplaceabilityStrategy	ClassificationType	hardwareElementReplaceabilityStrategy is a classification that identifies the replaceability strategy chosen for a specific customer and maintenance concept. **Note** Replaceability strategy can include information about the maintenance level at which the replacement task is to be performed.	HardwareElementRevision	S3000L UoF Hardware Element

Attribute Name	Type	Definition	Class Name	UoF
hardwareElementStructuralIndicator	ClassificationType	hardwareElementStructuralIndicator is a classification that defines how important the HardwareElement is to structural integrity. **Note** A Structural Item (SI) is part of the systems bodywork. **Note** A Structure Significant Item (SSI) is an item which has been identified by a selection process coming from a Preventive Maintenance Analysis (PMA) procedure eg, S4000P. For this item, a PMA will be performed. **Note** A Structural Detail (SD), also named Significant Detail in S4000P, is a limited area or a local spot of an SSI. In contrast to an SSI, the SD isn´t identified by its own part identifier (eg, its own part number).	HardwareElementRevision	S3000L UoF Hardware Element
hardwareElementType	ClassificationType	hardwareElementType is a classification that identifies further specialization for a HardwareElement. **Example** – Access point – Door – Electrical panel – Equipment – Panel – Slot	HardwareElement	S3000L UoF Hardware Element
hardwarePartConsumptionRate	PropertyType	hardwarePartConsumptionRate is a property that specifies the number of times that the part is replaced in 100 repairs of the next higher assembly.	HardwarePartAsDesignedSupportData	S3000L UoF Part Definition
hardwarePartElectromagneticIncompatible	umlBoolean	hardwarePartElectromagneticIncompatible is a classification that identifies the ability of an electrical HardwarePartAsDesigned to function satisfactorily in its electromagnetic environment without inadmissibly influencing this environment to which other HardwarePartAsDesigned belong.	HardwarePartAsDesignedDesignData	S3000L UoF Part Definition

Attribute Name	Type	Definition	Class Name	UoF
hardwarePartElectromagneticSensitive	umlBoolean	hardwarePartElectromagneticSensitive is a classification that specifies whether the HardwarePartAsDesigned can be subject to catastrophic failure, major characteristic change or performance degradation, due to influences of electromagnetic fields.	HardwarePartAsDesignedDesignData	S3000L UoF Part Definition
hardwarePartElectrostaticSensitive	umlBoolean	hardwarePartElectrostaticSensitive is a classification that specifies whether the HardwarePartAsDesigned can be subject to catastrophic failure, major characteristic change or performance degradation, due to influences of electrostatic fields and/or electrical charges.	HardwarePartAsDesignedDesignData	S3000L UoF Part Definition
hardwarePartEnvironmentalAspectInUseClass	DatedClassification	hardwarePartEnvironmentalAspectInUseClass is a classification that identifies environment aspects that must to be considered during use of and maintenance on the HardwarePartAsDesigned.	HardwarePartAsDesignedSupportData	S3000L UoF Part Definition
hardwarePartEnvironmentalAspectPlannedDisposalClass	DatedClassification	hardwarePartEnvironmentalAspectPlannedDisposalClass is a classification that identifies environment aspects that must to be considered during planned disposal of the HardwarePartAsDesigned.	HardwarePartAsDesignedSupportData	S3000L UoF Part Definition
hardwarePartFitmentRequirement	ClassificationType	hardwarePartFitmentRequirement is a classification that identifies if a HardwarePartAsDesigned cannot be fitted in its 'as supplied' state but must undergo some operation before or during installation.	HardwarePartAsDesignedDesignData	S3000L UoF Part Definition
hardwarePartHazardousClass	ClassificationType	hardwarePartHazardousClass is a classification that identifies to what extent a HardwarePartAsDesigned is capable of posing a significant risk to health, safety or property during transportation, handling, or storage.	HardwarePartAsDesignedDesignData	S3000L UoF Part Definition
hardwarePartLogisticsCategory	ClassificationType	hardwarePartLogisticsCategory is a classification that defines the role of the HardwarePartAsDesigned in the context of product support.	HardwarePartAsDesignedSupportData	S3000L UoF Part Definition

Attribute Name	Type	Definition	Class Name	UoF
hardwarePartMagneticSensitive	umlBoolean	hardwarePartMagneticSensitive is a classification that specifies whether the HardwarePartAsDesigned can be subject to catastrophic failure, major characteristic change or performance degradation, due to influences of magnetic fields.	HardwarePartAsDesignedDesignData	S3000L UoF Part Definition
hardwarePartMaintenanceSchedulingStart	ClassificationType	hardwarePartMaintenanceSchedulingStart is a classification that specifies when maintenance scheduling must be initiated for a realized part.	HardwarePartAsDesignedSupportData	S3000L UoF Part Definition
hardwarePartOperationalAuthorizedLife	AuthorizedLife	hardwarePartOperationalAuthorizedLife is an extended property that identifies the maximum usage limit for which an item can be operated, and upon reaching this limit, any further usage of the item must be re-authorized. **Example** – Calendar – Cycles – Hours – Landings	HardwarePartAsDesignedDesignData	S3000L UoF Part Definition
hardwarePartRadiationSensitive	umlBoolean	hardwarePartRadiationSensitive is a classification that specifies whether the HardwarePartAsDesigned can be subject to catastrophic failure, major characteristic change or performance degradation, due to influences of radioactive radiation.	HardwarePartAsDesignedDesignData	S3000L UoF Part Definition
hardwarePartRepairability	ClassificationType	hardwarePartRepairability is a classification that identifies the extent to which the HardwarePartAsDesigned is repairable from a technical perspective, independent of customer maintenance concepts.	HardwarePartAsDesignedSupportData	S3000L UoF Part Definition
hardwarePartRepairabilityStrategy	ClassificationType	hardwarePartRepairabilityStrategy is a classification that specifies if a repairable HardwarePartAsDesigned is to repaired and optionally also at what maintenance level or maintenance facility. Note hardwarePartRepairabilityStrategy is often defined in the context of a specific customer and maintenance concept (Refer to UoF Applicability Statement).	HardwarePartAsDesignedSupportData	S3000L UoF Part Definition

Attribute Name	Type	Definition	Class Name	UoF
hardwarePartScrapRate	NumericalPropertyType	hardwarePartScrapRate is a property that defines the fraction of repairable units which, when removed from service, will be found to be beyond economic repair and therefore have to be scrapped.	HardwarePartAsDesignedSupportData	S3000L UoF Part Definition
hardwarePartSize	ThreeDimensional	hardwarePartSize is an extended property that specifies the spatial magnitudes for a HardwarePartAsDesigned. **Note** hardwarePartSize must exclude packaging and other external factors.	HardwarePartAsDesignedDesignData	S3000L UoF Part Definition
hardwarePartSupportFamilyClass	ClassificationType	hardwarePartSupportFamilyClass is a classification that generalizes the HardwarePartAsDesigned from a support analysis perspective. **Note** hardwarePartSupportFamilyClass can be used to organize hardware parts that share similar support characteristics and/or mainitenance solution. **Example** – Gearbox	HardwarePartAsDesignedSupportData	S3000L UoF Part Definition
hardwarePartTraceabilityClass	ClassificationType	hardwarePartTraceabilityClass is a classification that specifies the way the HardwarePartAsDesigned is to be tracked and traced when entered into service.	HardwarePartAsDesignedSupportData	S3000L UoF Part Definition
hardwarePartWasteProductsInUseDisposalDescription	DescriptorType	hardwarePartWasteProductsInUseDisposalDescription is a description that gives more information on how waste products for an individual HardwarePartAsDesigned need to be managed when the part is disposed of according to the procedure during use or after being used.	HardwarePartAsDesignedSupportData	S3000L UoF Part Definition

Attribute Name	Type	Definition	Class Name	UoF
hardwarePartWasteProductsPlannedDisposalDescription	DescriptorType	hardwarePartWasteProductsPlannedDisposalDescription is a description that gives more information on how waste products need to be managed when the entire population of a HardwarePartAsDesigned is disposed of according to the procedure for planned disposal.	HardwarePartAsDesignedSupportData	S3000L UoF Part Definition
hardwarePartWeight	PropertyType	hardwarePartWeight is a property that specifies the mass for a HardwarePartAsDesigned. **Note** hardwarePartWeight must exclude packaging and other external factors.	HardwarePartAsDesignedDesignData	S3000L UoF Part Definition
informationCode	umlString	informationCode is a string of characters that represents the information code attribute of the data module code. **Note** An informationCode must be created in accordance with the rules defined in S1000D.	S1000DDataModule	S3000L UoF Document
informationCodeVariant	umlString	informationCodeVariant is a string of characters that represents the information code variant attribute of the data module code. **Note** An informationCodeVariant must be created in accordance with the rules defined in S1000D.	S1000DDataModule	S3000L UoF Document
infrastructureCompliance Date	DateType	infrastructureComplianceDate is a date that defines when infrastructure compliance was declared.	InfrastructureCompliance	S3000L UoF Facility
infrastructureCompliance Description	DescriptorType	infrastructureComplianceDescription is a description that gives more information on compliance fulfillment.	InfrastructureCompliance	S3000L UoF Facility
infrastructureCompliance Level	ClassificationType	infrastructureComplianceLevel is a classification that specifies the degree of compliance.	InfrastructureCompliance	S3000L UoF Facility

Attribute Name	Type	Definition	Class Name	UoF
inServiceOptimizationAnalysisName	NameType	inServiceOptimizationAnalysisName is a name by which the InServiceOptimizationAnalysis is known and can be easily referenced.	InServiceOptimizationAnalysis	S3000L UoF In Service Optimization Analysis
inServiceOptimizationAnalysisRationale	DescriptorType	inServiceOptimizationAnalysisRationale is a description that gives more information on the reason for the defined InServiceOptimizationAnalysis.	InServiceOptimizationAnalysisRevision	S3000L UoF In Service Optimization Analysis
inServiceOptimizationAnalysisRevisionDate	DateType	inServiceOptimizationAnalysisRevisionDate is a date that specifies when an InServiceOptimizationAnalysis was revised.	InServiceOptimizationAnalysisRevision	S3000L UoF In Service Optimization Analysis
inServiceOptimizationAnalysisRevisionIdentifier	IdentifierType	inServiceOptimizationAnalysisRevisionIdentifier is an identifier that establishes a unique designator for an InServiceOptimizationAnalysisRevision and to differentiate it from other instances of InServiceOptimizationAnalysisRevision.	InServiceOptimizationAnalysisRevision	S3000L UoF In Service Optimization Analysis
inServiceOptimizationAnalysisRevisionRationale	DescriptorType	inServiceOptimizationAnalysisRevisionRationale is a description that gives more information on the justification for revising the defined InServiceOptimizationAnalysis.	InServiceOptimizationAnalysisRevision	S3000L UoF In Service Optimization Analysis
inServiceOptimizationAnalysisRevisionStatus	StateType	inServiceOptimizationAnalysisRevisionStatus is a state that identifies the maturity of an InServiceOptimizationAnalysisRevision.	InServiceOptimizationAnalysisRevision	S3000L UoF In Service Optimization Analysis
inServiceOptimizationAnalysisStepDescription	DescriptorType	inServiceOptimizationAnalysisStepDescription is a description that gives information on actions taken, information gathered, decisions and conclusions made during the InServiceOptimizationAnalysisStep.	InServiceOptimizationAnalysisStep	S3000L UoF In Service Optimization Analysis
inServiceOptimizationAnalysisStepIdentifier	IdentifierType	inServiceOptimizationAnalysisStepIdentifier is an identifier that establishes a unique designator for an InServiceOptimizationAnalysisStep and to differentiate it from other instances of InServiceOptimizationAnalysisStep.	InServiceOptimizationAnalysisStep	S3000L UoF In Service Optimization Analysis

Attribute Name	Type	Definition	Class Name	UoF
inServiceOptimizationAnalysisStepQuestionAnswer	umlBoolean	inServiceOptimizationAnalysisStepQuestionAnswer is a boolean that specifies a TRUE or FALSE conclusion to the associated question.	QuestionInServiceOptimizationAnalysisStep	S3000L UoF In Service Optimization Analysis
inServiceOptimizationAnalysisSummaryDescription	DescriptorType	inServiceOptimizationAnalysisSummaryDescription is a description that gives summarized information on decisions and conclusions made during the InServiceOptimizationAnalysis.	InServiceOptimizationAnalysisRevision	S3000L UoF In Service Optimization Analysis
itemLocationCode	umlString	itemLocationCode is a string of characters that represents the item location code attribute of the data module code. **Note** An itemLocationCode must be created in accordance with the rules defined in S1000D.	S1000DDataModule	S3000L UoF Document
latitude	umlString	latitude is a string of characters that represents a geographic coordinate specifying the north-south position of a point. **Example** – 39.5693900 – 39°34.1634' N – 39°34'09" N	GlobalPosition	S3000L UoF Location
learnCode	umlString	learnCode is a string of characters that represents the learn code attribute of the data module code. **Note** A learnCode must be created in accordance with the rules defined in S1000D.	S1000DDataModule	S3000L UoF Document
learnEventCode	umlString	learnEventCode is a string of characters that represents the learn event code attribute of the data module code. **Note** A learnEventCode must be created in accordance with the rules defined in S1000D.	S1000DDataModule	S3000L UoF Document

Attribute Name	Type	Definition	Class Name	UoF
locationRelationshipType	ClassificationType	locationRelationshipType is a classification that identifies the meaning of the established relationship. **Example** – located in – located next to	LocationRelationship	S3000L UoF Location
longitude	umlString	longitude is a string of characters that represents a geographic coordinate specifying the east–west position of a point. **Example** – 2.6502400° – 2°39.0144' E – 2°39'00" E	GlobalPosition	S3000L UoF Location
lsaFailureModeGroupDescription	DescriptorType	lsaFailureModeGroupDescription is a description that gives more information on the LSAFailureModeGroup.	LSAFailureModeGroup	S3000L UoF LSA Failure Mode Group
lsaFailureModeGroupDistributionRatio	PropertyType	lsaFailureModeGroupDistributionRatio is a property that identifies the occurrence fraction of the LSAFailureModeGroup in relation to the entire population of LSAFailureModeGroups identified for the AnalysisActivityItem.	LSAFailureModeGroup	S3000L UoF LSA Failure Mode Group
lsaFailureModeGroupIdentifier	IdentifierType	lsaFailureModeGroupIdentifier is an identifier that establishes a unique designator for a LSAFailureModeGroup and to differentiate it from other instances of LSAFailureModeGroup.	LSAFailureModeGroup	S3000L UoF LSA Failure Mode Group
lsaFailureModeGroupName	NameType	lsaFailureModeGroupName is a name by which the LSAFailureModeGroup is known and can be easily referenced.	LSAFailureModeGroup	S3000L UoF LSA Failure Mode Group
lsaFailureModeGroupTaskRequirementPurpose	ClassificationType	lsaFailureModeGroupTaskRequirementPurpose is a classification that identifies the role of the referred TaskRequirement in the context of the LSAFailureModeGroup.	LSAFailureModeGroupTaskRequirement	S3000L UoF LSA Failure Mode Group
maintenanceFacilityType	ClassificationType	maintenanceFacilityType is a classification that identifies further specialization for a MaintenanceFacility.	MaintenanceFacility	S3000L UoF Facility

Attribute Name	Type	Definition	Class Name	UoF
maintenanceLevelCapabilityDescription	DescriptorType	maintenanceLevelCapabilityDescription is a description that gives more information on the ability to perform maintenance based on availability of support resources and environmental conditions. **Note** The defined abilities are the basis for determining the functions to be accomplished at the defined maintenance level. **Note** Support resources include eg, personnel and skills, special facilities and support equipment, etc.	MaintenanceLevel	S3000L UoF Product Usage Context
maintenanceLevelIdentifier	IdentifierType	maintenanceLevelIdentifier is an identifier that establishes a unique designator for a MaintenanceLevel and to differentiate it from other instances of MaintenanceLevel.	MaintenanceLevel	S3000L UoF Product Usage Context
maintenanceLevelName	NameType	maintenanceLevelName is a name by which the MaintenanceLevel is known and can be easily referenced.	MaintenanceLevel	S3000L UoF Product Usage Context
maintenanceSignificantOrRelevant	ClassificationType	maintenanceSignificantOrRelevant is a classfication that identifies whether a BreakdownElement requires maintenance activities or not. **Note** A maintenance relevant item is an item which can be repaired or replaced as a result of failure or damage. **Note** A maintenance significant item is an item which has been identified by a selection process coming from a Preventive Maintenance Analysis (PMA) eg, S4000P. For this type of item, a preventive maintenance task will be documented.	BreakdownElementRevision	S3000L UoF Breakdown Structure

Attribute Name	Type	Definition	Class Name	UoF
materialItemCategoryCode	umlString	materialItemCategoryCode is a string of characters that represents the material item category code attribute of the data module code. **Note** A materialItemCategoryCode must be created in accordance with the rules defined in S1000D.	S1000DDataModule	S3000L UoF Document
measurementPointDefinitionDescription	DescriptorType	measurementPointDefinitionDescription is a description that gives more information on the MeasurementPointDefinition.	MeasurementPointDefinition	S3000L UoF Failure Mode Symptom
measurementPointDefinitionIdentifier	IdentifierType	measurementPointDefinitionIdentifier is an identifier that establishes a unique designator for a MeasurementPointDefinition and to differentiate it from other instances of MeasurementPointDefinition.	MeasurementPointDefinition	S3000L UoF Failure Mode Symptom
measurementPointDefinitionName	NameType	measurementPointDefinitionName is a name by which the MeasurementPointDefinition is known and can be easily referenced.	MeasurementPointDefinition	S3000L UoF Failure Mode Symptom
measurementPointDefinitionType	ClassificationType	measurementPointDefinitionType is a classification that identifies further specialization of MeasurementPointDefinition. **Example** – Gauge	MeasurementPointDefinition	S3000L UoF Failure Mode Symptom
measurementPointDegradedStateDefinitionAssertValue	PropertyType	measurementPointDegradedStateDefinitionAssertValue is a property that defines the value to be evaluated as part of the MeasurementPointDegradedStateDefinition.	MeasurementPointDegradedStateDefinition	S3000L UoF Time Limit
measurementPointDegradedStateDefinitionAssertValueComparisonOperator	ClassificationType	measurementPointDegradedStateDefinitionAssertValueComparisonOperator is a classification that identifies the comparison operator against which a recorded value is to be evaluated against the measurementPointDegradedStateDefinitionAssertValue.	MeasurementPointDegradedStateDefinition	S3000L UoF Time Limit

Attribute Name	Type	Definition	Class Name	UoF
measurementPointFaultStateDefinitionAssertValue	PropertyType	measurementPointFaultStateDefinitionAssertValue is a property that defines the value to be evaluated as part of the MeasurementPointFaultStateDefinition.	MeasurementPointFaultStateDefinition	S3000L UoF Failure Mode Symptom
measurementPointFaultStateDefinitionAssertValueComparisonOperator	ClassificationType	measurementPointFaultStateDefinitionAssertValueComparisonOperator is a classification that identifies the comparison operator against which a recorded value is to be evaluated against the measurementPointFaultStateDefinitionAssertValue.	MeasurementPointFaultStateDefinition	S3000L UoF Failure Mode Symptom
messageContentStatus	StateType	messageContentStatus is a state that identifies the quality assurance status of the message content.	Message	S3000L UoF Message
messageContentType	ClassificationType	messageContentType is a classification that characterizes the information included in the message content.	Message	S3000L UoF Message
messageCreationDateTime	DateTimeType	messageCreationDateTime is a date and time that defines when the Message was generated.	Message	S3000L UoF Message
messageIdentifier	IdentifierType	messageIdentifier is an identifier that establishes a unique designator for a Message and allows it to be differentiated from other instances of Message.	Message	S3000L UoF Message
messageLanguage	ClassificationType	messageLanguage is a classification that identifies the language of the information in the message content.	Message	S3000L UoF Message
messagePartyType	ClassificationType	messagePartyType is a classification that identifies the role of the associated MessagePartyItem. **Example** – Receiver – Sender	MessageParty	S3000L UoF Message
messageRelationshipType	ClassificationType	messageRelationshipType is a classification that characterizes the relationship that is established between two Messages.	MessageRelationship	S3000L UoF Message

Attribute Name	Type	Definition	Class Name	UoF
modelIdentificationCode	umlString	modelIdentificationCode is a string of characters that represents the model identification code attribute of the data module code. **Note** A modelIdentificationCode must be created in accordance with the rules defined in S1000D.	S1000DDataModule	S3000L UoF Document
			S1000DPublicationModule	S3000L UoF Document
nonConformanceDescription	DescriptorType	nonConformanceDescription is a description that gives more information on how the EffectiveOnProductConfigurationItem does not comply with its requirements.	NonConformanceData	S3000L UoF Product Design Configuration
nonConformanceRestriction	DescriptorType	nonConformanceRestriction is a description that gives more information on how the use of the related EffectiveOnProductConfigurationItem restricts the specified capabilities of the AllowedProductConfiguration in which it is contained.	NonConformanceData	S3000L UoF Product Design Configuration
nonConformanceType	ClassificationType	nonConformanceType is a classification that identifies in which way the EffectiveOnProductConfigurationItem does not comply with its requirements.	NonConformanceData	S3000L UoF Product Design Configuration
numberOfOperatingLocations	PropertyType	numberOfOperatingLocations is a property that specifies the number of locations of the OperatingLocationType that need to be considered when dimensioning for the support solution.	OperatingLocationType	S3000L UoF Product Usage Context
operatingBaseType	ClassificationType	operatingBaseType is a classification that identifies further specialization for a OperatingBase.	OperatingBase	S3000L UoF Facility
operatingLocationTypeDescription	DescriptorType	operatingLocationTypeDescription is a description that gives more information on the OperatingLocationType, including the environmental conditions to be expected.	OperatingLocationType	S3000L UoF Product Usage Context

Attribute Name	Type	Definition	Class Name	UoF
operatingLocationTypeIdentifier	IdentifierType	operatingLocationTypeIdentifier is an identifier that establishes a unique designator for an OperatingLocationType and to differentiate it from other instances of OperatingLocationType.	OperatingLocationType	S3000L UoF Product Usage Context
operatingLocationTypeName	NameType	operatingLocationTypeName is a name by which the OperatingLocationType is known and can be easily referenced.	OperatingLocationType	S3000L UoF Product Usage Context
operatingRequirementAtOperatingLocation	PropertyType	operatingRequirementAtOperatingLocation is a property that specifies the (annual) operating requirement per operating location and contracted ProductVariant. **Note** Annual operating requirements are typically based on operating hours, but other measurement bases are also possible.	ContractedProductVariantAtOperatingLocation	S3000L UoF Product Usage Context
operatingRequirementAtOperatingLocationType	PropertyType	operatingRequirementAtOperatingLocationType is a property that specifies the (annual) operating requirement per OperatingLocationType and contracted ProductVariant. **Note** Annual operating requirements are typically based on operating hours, but other measurement bases are also possible.	ContractedProductVariantAtOperatingLocationType	S3000L UoF Product Usage Context
packagedTask	umlBoolean	packagedTask is a boolean that specifies if the Task is created in order to group a set of defined Tasks for a specific purpose. **Note** Grouping of Tasks that can be performed to support maintenance planning and scheduling activities. **Example** – 1000 flight hours overhaul	RectifyingTask	S3000L UoF Task
partAsDesignedPartsListRelationshipType	ClassificationType	partAsDesignedPartsListRelationshipType is a classification that identifies the meaning of the established relationship.	PartAsDesignedPartsListRelationship	S3000L UoF Part Definition

Attribute Name	Type	Definition	Class Name	UoF
partDemilitarizationClass	DatedClassification	partDemilitarizationClass is a classification defining special measures to be taken when a PartAsDesigned is being disposed of. **Note** Special measures can include rendering an item useless for military purposes or destroy any indications of military purposes or performance characteristics.	PartAsDesignedControlledItemData	S3000L UoF Part Definition
partIdentifier	IdentifierType	partIdentifier is an identifier that establishes a unique designator for a PartAsDesigned and to differentiate it from other instances of PartAsDesigned. **Note** Part identification includes drawing, model, type, or source controlling numbers. **Example** - '12345-501'	PartAsDesigned	S3000L UoF Part Definition
partMaturityClass	DatedClassification	partMaturityClass is a classification that defines the maturity of the PartAsDesigned in order to determine the certainty by which its characteristics can be valued.	PartAsDesignedSupportData	S3000L UoF Part Definition
partName	NameType	partName is a name by which the PartAsDesigned is known and can be easily referenced.	PartAsDesigned	S3000L UoF Part Definition
partObsolescenceRiskAssessment	DescriptorType	partObsolescenceRiskAssessment is a description of the risk associated with loss of the PartAsDesigned in the supply chain.	PartAsDesignedSupportData	S3000L UoF Part Definition
partsListEntryIdentifier	IdentifierType	partsListEntryIdentifier is an identifier that establishes a unique designator for a PartAsDesignedPartsListEntry and to differentiate it from other instances of PartAsDesignedPartsListEntry.	PartAsDesignedPartsListEntry	S3000L UoF Part Definition
partsListEntryQuantity	PropertyType	partsListEntryQuantity is a property that specifies the amount of the PartAsDesigned used in its parent PartAsDesignedPartsListRevision.	PartAsDesignedPartsListEntry	S3000L UoF Part Definition

Attribute Name	Type	Definition	Class Name	UoF
partsListRevisionDate	DateType	partsListRevisionDate is a date that specifies when the PartAsDesignedPartsList was revised.	PartAsDesignedPartsListRevision	S3000L UoF Part Definition
partsListRevisionIdentifier	IdentifierType	partsListRevisionIdentifier is an identifier that establishes a unique designator for a PartAsDesignedPartsListRevision and to differentiate it from other instances of PartAsDesignedPartsListRevision for the same partsListType.	PartAsDesignedPartsListRevision	S3000L UoF Part Definition
partsListRevisionRationale	DescriptorType	partsListRevisionRationale is a description that gives more information on the justification for revising the PartAsDesignedPartsList.	PartAsDesignedPartsListRevision	S3000L UoF Part Definition
partsListRevisionStatus	StateType	partsListRevisionStatus is a state that identifies the maturity of a PartAsDesignedPartsListRevision.	PartAsDesignedPartsListRevision	S3000L UoF Part Definition
partsListType	ClassificationType	partsListType is a classification that identifies the context and intended use of the PartAsDesignedPartsList.	PartAsDesignedPartsList	S3000L UoF Part Definition
partSpecialHandlingRequirement	DescriptorType	partSpecialHandlingRequirement is a description that gives more information on requirements for special handling of the PartAsDesigned.	PartAsDesignedDesignData	S3000L UoF Part Definition
performanceParameterCalculationMethod	DescriptorType	performanceParameterCalculationMethod is a description that gives more information on the method by which the performanceParameterValue has been derived.	PerformanceParameterValueGroup	S3000L UoF Performance Parameter
performanceParameterRevisionDate	DateType	performanceParameterRevisionDate is a date that specifies when a PerformanceParameterRevision was defined.	PerformanceParameterRevision	S3000L UoF Performance Parameter
performanceParameterRevisionIdentifier	IdentifierType	performanceParameterRevisionIdentifier is an identifier that establishes a unique designator for a PerformanceParameterRevision and to differentiate it from other instances of PerformanceParameterRevision.	PerformanceParameterRevision	S3000L UoF Performance Parameter

Attribute Name	Type	Definition	Class Name	UoF
performanceParameterRevisionRationale	DescriptorType	performanceParameterRevisionRationale is a description that gives more information on the justification for revising the defined PerformanceParameter and its values.	PerformanceParameterRevision	S3000L UoF Performance Parameter
performanceParameterRevisionStatus	StateType	performanceParameterRevisionStatus is a state that identifies the maturity of a PerformanceParameterRevision.	PerformanceParameterRevision	S3000L UoF Performance Parameter
performanceParameterType	ClassificationType	performanceParameterType is a classification that identifies the type of PerformanceParameter being exchanged.	PerformanceParameter	S3000L UoF Performance Parameter
performanceParameterValue	PropertyType	performanceParameterValue is a property that represents a value which is determined for the PerformanceParameter.	PerformanceParameterValueGroup	S3000L UoF Performance Parameter
performanceParameterValueFraction	PropertyType	performanceParameterValueFraction is a property that represents the fraction of all occurrences related to a specified PerformanceParameter that must be within the limit of the defined performanceParameterValue. **Example** – A customer requirement is that 98% of all replacement tasks must be performed below a specified value of two hours (= maximum replacement time)	PerformanceParameterValueGroup	S3000L UoF Performance Parameter
performanceParameterValueLimitQualifier	ClassificationType	performanceParameterValueLimitQualifier is a classification that specifies a directed limit to be applied when testing a value against the defined PerformanceParameter.	PerformanceParameter	S3000L UoF Performance Parameter
physicalReplaceability	ClassificationType	physicalReplaceability is a classification which identifies whether a part is interchangeable in a specific assembly.	PartAsDesignedPartsListEntry	S3000L UoF Part Definition
physicalReplaceabilityStrategy	ClassificationType	physicalReplaceabilityStrategy is a classification that identifies the replaceability strategy chosen for a specific customer and maintenance concept. **Note** Replaceability strategy can include information about the maintenance level, at which the replacement task is to be performed.	PartAsDesignedPartsListEntry	S3000L UoF Part Definition

Attribute Name	Type	Definition	Class Name	UoF
postalCode	umlString	postalCode is a string of characters that represents a short code used by the postal service to identify a geographical area.	StreetAddress	S3000L UoF Location
productIdentifier	IdentifierType	productIdentifier is an identifier that establishes a unique designator for a Product and to differentiate it from other instances of Product.	Product	S3000L UoF Product and Project
productName	NameType	productName is a name by which the Product is known and can be easily referenced.	Product	S3000L UoF Product and Project
productUsagePhaseDescription	DescriptorType	productUsagePhaseDescription is a description that gives more information on the ProductUsagePhase.	ProductUsagePhase	S3000L UoF Product Usage Phase
productUsagePhaseDuration	PropertyType	productUsagePhaseDuration is a property that specifies the average time that the ProductUsagePhaseItem will remain in the specified state.	ProductUsagePhase	S3000L UoF Product Usage Phase
productUsagePhaseName	NameType	productUsagePhaseName is a name by which the ProductUsagePhase is known and can be easily referenced.	ProductUsagePhase	S3000L UoF Product Usage Phase
productUsagePhaseRelationshipType	ClassificationType	productUsagePhaseRelationshipType is a classification that identifies the meaning of the established relationship.	ProductUsagePhaseRelationship	S3000L UoF Product Usage Phase
productVariantIdentifier	IdentifierType	productVariantIdentifier is an identifier that establishes a unique designator for a ProductVariant and to differentiate it from other instances of ProductVariant.	ProductVariant	S3000L UoF Product and Project
productVariantName	NameType	productVariantName is a name by which the ProductVariant is known and can be easily referenced.	ProductVariant	S3000L UoF Product and Project
projectIdentifier	IdentifierType	projectIdentifier is an identifier that establishes a unique designator for a Project and to differentiate it from other instances of Project.	Project	S3000L UoF Product and Project
projectName	NameType	projectName is a name by which the Project is known and can be easily referenced.	Project	S3000L UoF Product and Project

Attribute Name	Type	Definition	Class Name	UoF
proposedTrainingMethod	ClassificationType	proposedTrainingMethod is a classification that suggests the way in which the additional learning can be acquired.	AdditionalTrainingNeed	S3000L UoF Task Resource
publicationModuleIssueInWorkNumber	umlString	publicationModuleIssueInWorkNumber is a string of characters used for monitoring and control of intermediate drafts of S1000DPublicationModuleIssue. **Note** A publicationModuleIssueInWorkNumber must be created in accordance with the rules defined in S1000D.	S1000DPublicationModuleIssue	S3000L UoF Document
publicationModuleIssueLanguage	ClassificationType	publicationModuleIssueLanguage is a classification that identifies the language used to produce the content of the S1000DPublicationModuleIssue. **Note** A publicationModuleIssueLanguage must be created in accordance with the rules defined in S1000D.	S1000DPublicationModuleIssue	S3000L UoF Document
publicationModuleIssueLanguageCountry	ClassificationType	publicationModuleIssueLanguageCountry is a classification that identifies the country where the language, identified by publicationModuleIssueLanguage, is spoken. **Note** A publicationModuleIssueLanguageCountry must be created in accordance with the rules defined in S1000D.	S1000DPublicationModuleIssue	S3000L UoF Document
publicationModuleIssueNumber	umlString	publicationModuleIssueNumber is a string of characters used to identify the release number of the S1000DPublicationModuleIssue. **Note** A publicationModuleIssueNumber must be created in accordance with the rules defined in S1000D.	S1000DPublicationModuleIssue	S3000L UoF Document

Attribute Name	Type	Definition	Class Name	UoF
publicationModuleIssuer	umlString	publicationModuleIssuer is a string of characters that represents the issuing authority attribute of the publication module code. **Note** A publicationModuleIssuer must be created in accordance with the rules defined in S1000D.	S1000DPublicationModule	S3000L UoF Document
publicationModuleNumber	umlString	publicationModuleNumber is a string of characters that represents the number of the publication module attribute of the publication module code. **Note** A publicationModuleNumber must be created in accordance with the rules defined in S1000D.	S1000DPublicationModule	S3000L UoF Document
publicationModuleVolume	umlString	publicationModuleVolume is a string of characters that represents the volume of the publication attribute of the publication module code. **Note** A publicationModuleVolume must be created in accordance with the rules defined in S1000D.	S1000DPublicationModule	S3000L UoF Document
quantityOfContainedSubstance	PropertyType	quantityOfContainedSubstance is a property that identifies the amount of the substance included in the HardwarePartAsDesigned.	ContainedSubstance	S3000L UoF Part Definition
quantityOfProductVariantAtOperatingLocation	PropertyType	quantityOfProductVariantAtOperatingLocation is a property that specifies the number of serialized items, of a specific ProductVariant, that is to be operated at a specific operating location.	ContractedProductVariantAtOperatingLocation	S3000L UoF Product Usage Context
quantityOfProductVariantAtOperatingLocationType	PropertyType	quantityOfProductVariantAtOperatingLocationType is a property that specifies the number of serialized items, of a specific ProductVariant, that is to be operated at a specific OperatingLocationType.	ContractedProductVariantAtOperatingLocationType	S3000L UoF Product Usage Context

Attribute Name	Type	Definition	Class Name	UoF
referencedDigitalFileJustification	DescriptorType	referencedDigitalFileJustification is a phrase that provides more information on the reason why the DigitalFile is referenced. **Example** – A video showing the task execution.	ReferencedDigitalFile	S3000L UoF Digital File
referencedDocumentPortion	DescriptorType	referencedDocumentPortion is a description that provides a reference to the portion of a Document which is of interest in a specific usage.	ReferencedDocument	S3000L UoF Document
referencedDocumentRole	ClassificationType	referencedDocumentRole is a classification that identifies the function of the established relationship. **Example** – Design document reference – Directive – Document reference – Drawing reference – Source – Verification	ReferencedDocument	S3000L UoF Document
referenceDesignator	IdentifierType	referenceDesignator is an identifier that establishes a unique designator for a location within the overall Product, and to differentiate it from other locations. **Note** Reference designators serve as a cross reference between parts contained in wiring diagrams, hydraulic systems etc. and eg, the Illustrated Parts Data (IPD).	BreakdownElementUsageInBreakdown	S3000L UoF Breakdown Structure
			PartAsDesignedPartsListEntry	S3000L UoF Part Definition
referencedOrganizationRole	ClassificationType	referencedOrganizationRole is a classification that identifies the function of the established relationship.	ReferencedOrganization	S3000L UoF Organization Assignment
remarkText	DescriptorType	remarkText is a description that provides the text of the additional information.	Remark	S3000L UoF Remark

Attribute Name	Type	Definition	Class Name	UoF
remarkType	ClassificationType	remarkType is a classification that defines the purpose of the Remark. **Example** – Internal note – Technical fact	Remark	S3000L UoF Remark
resourceSpecificationDescription	DescriptorType	resourceSpecificationDescription is a description that gives more information on the characteristics that a part realization must fulfill in order to qualify as a possible resource.	ResourceSpecificationRevision	S3000L UoF Resource Specification
resourceSpecificationIdentifier	IdentifierType	resourceSpecificationIdentifier is an identifier that establishes a unique designator for a ResourceSpecification and to differentiate it from other instances of ResourceSpecification.	ResourceSpecification	S3000L UoF Resource Specification
resourceSpecificationName	NameType	resourceSpecificationName is a name by which the ResourceSpecification is known and can be easily referenced.	ResourceSpecificationRevision	S3000L UoF Resource Specification
resourceSpecificationRevisionDate	DateType	resourceSpecificationRevisionDate is a date that specifies when the ResourceSpecification was revised.	ResourceSpecificationRevision	S3000L UoF Resource Specification
resourceSpecificationRevisionIdentifier	IdentifierType	resourceSpecificationRevisionIdentifier is an identifier that establishes a unique designator for a ResourceSpecificationRevision and to differentiate it from other instances of ResourceSpecificationRevision.	ResourceSpecificationRevision	S3000L UoF Resource Specification
resourceSpecificationRevisionRationale	DescriptorType	resourceSpecificationRevisionRationale is a description that gives more information on the justification for revising the ResourceSpecification.	ResourceSpecificationRevision	S3000L UoF Resource Specification
resourceSpecificationRevisionStatus	StateType	resourceSpecificationRevisionStatus is a state that identifies the maturity of a ResourceSpecificationRevision.	ResourceSpecificationRevision	S3000L UoF Resource Specification
resourceSpecificationType	ClassificationType	resourceSpecificationType is a classification that identifies further specialization for a ResourceSpecification.	ResourceSpecification	S3000L UoF Resource Specification

Attribute Name	Type	Definition	Class Name	UoF
samplingMethodDescription	DescriptorType	samplingMethodDescription is a description that gives more information on the way in which the sample must be selected.	SamplingDefinition	S3000L UoF Time Limit
samplingValue	PropertyType	samplingValue is a property that specifies the number or fraction of the total population to be selected.	SamplingDefinition	S3000L UoF Time Limit
securityClassificationAuthority	Organization	securityClassificationAuthority identifies the Organization that is the authoritative source for the defined SecurityClassification.	SecurityClassification	S3000L UoF Security Classification
securityClassificationDate	DateType	securityClassificationDate is a date when the security classification is declared.	SecurityClassification	S3000L UoF Security Classification
securityClassValue	NameType	securityClassValue is a name that defines the level of confidentiality. **Example** – Company confidential – Confidential – Restricted – Secret – Top secret – Unclassified	SecurityClass	S3000L UoF Security Classification
skillCode	IdentifierType	skillCode is an identifier that establishes a unique designator for a Skill and to differentiate it from other instances of Skill.	Skill	S3000L UoF Competence Definition
skillLevelDescription	DescriptorType	skillLevelDescription is a description that gives more information on a proficiency.	SkillLevel	S3000L UoF Competence Definition
skillLevelName	NameType	skillLevelName is a name that uniquely establishes a proficiency.	SkillLevel	S3000L UoF Competence Definition

Attribute Name	Type	Definition	Class Name	UoF
softwareElementLoadingStrategy	ClassificationType	softwareElementLoadingStrategy is a classification that identifies the loading strategy chosen for a specific customer and maintenance concept. **Note** Software loading strategy can be used to include information about the maintenance level at which the software loading task is to be performed.	SoftwareElementRevision	S3000L UoF Software Element
softwareElementModificationFrequency	PropertyType	softwareElementModificationFrequency is a property that defines the expected frequency with which the SoftwarePartAsDesigned which realizes this SoftwareElementRevision will be modified. **Example** – 3 months – 5 years	SoftwareElementRevision	S3000L UoF Software Element
softwareElementSize	PropertyType	softwareElementSize is a property that defines the size of the SoftwarePartAsDesigned which realizes this SoftwareElementRevision. **Example** – 10.000 lines of code - estimated – 23.5 Mbytes - executable – 800 Kbytes - contracted	SoftwareElementRevision	S3000L UoF Software Element
softwareElementType	ClassificationType	softwareElementType is a classification that identifies further specialization for a SoftwareElement.	SoftwareElement	S3000L UoF Software Element
softwarePartSize	PropertyType	softwarePartSize is a property that specifies the volume which the software will occupy on a digital storage media.	SoftwarePartAsDesignedDesignData	S3000L UoF Part Definition
softwarePartType	ClassificationType	softwarePartType is a classification that identifies how the SoftwarePartAsDesigned is designed to be installed on the related HardwarePartAsDesigned.	SoftwarePartAsDesignedDesignData	S3000L UoF Part Definition

Attribute Name	Type	Definition	Class Name	UoF
sourceDocumentPortion	DescriptorType	sourceDocumentPortion is a description that gives more information on relevant portions of the related Document.	SubtaskSourceDocument	S3000L UoF Task
specialEventDefinitionCauseCategory	ClassificationType	specialEventDefinitionCauseCategory is a classification that identifies a generalization that organize special events into an overarching special event taxonomy.	SpecialEventDefinition	S3000L UoF Special Event
specialEventDefinitionDecription	DescriptorType	specialEventDefinitionDecription is a description that gives more information on the SpecialEventDefinition.	SpecialEventDefinition	S3000L UoF Special Event
specialEventDefinitionName	NameType	specialEventDefinitionName is a name by which the SpecialEventDefinition is known and can be easily referenced.	SpecialEventDefinition	S3000L UoF Special Event
specialEventDefinitionOccurrenceValue	PropertyType	specialEventDefinitionOccurrenceValue is a property that quantifies how often a defined special event is likely to occur.	SpecialEventDefinition	S3000L UoF Special Event
specialEventOccurenceRatio	PropertyType	specialEventOccurenceRatio identifies the fraction of an individual SpecialEventDefinition in relation to the entire population of special events identified for the Product that will occur in relation to the associated ProductUsagePhase.	SpecialEventProductUsagePhaseOccurrence	S3000L UoF Special Event
streetName	NameType	streetName is the name by which a road is officially known and can be easily referenced. **Example** – E-2561 Road – Main Street	StreetAddress	S3000L UoF Location
streetNumber	umlString	streetNumber is a string of characters that represents the position along a street. **Example** – 4 – Km 35.5	StreetAddress	S3000L UoF Location

Attribute Name	Type	Definition	Class Name	UoF
substanceCharacteristicsRecordingDate	DateType	substanceCharacteristicsRecordingDate is a date that defines when a SubstanceDefinition was last updated.	SubstanceDefinition	S3000L UoF Part Definition
substanceDescription	DescriptorType	substanceDescription is a description that gives more information on the substance.	SubstanceDefinition	S3000L UoF Part Definition
substanceIdentifier	IdentifierType	substanceIdentifier is an identifier that establishes a unique designator for a SubstanceDefinition and to differentiate it from other instances of SubstanceDefinition.	SubstanceDefinition	S3000L UoF Part Definition
substanceName	NameType	substanceName is a name by which the SubstanceDefinition is known and can be easily referenced.	SubstanceDefinition	S3000L UoF Part Definition
substanceRiskDescription	DescriptorType	substanceRiskDescription is a description that specifies risks associated with the substance.	SubstanceDefinition	S3000L UoF Part Definition
substanceRiskFactor	ClassificationType	substanceRiskFactor is a classification which identifies possible risks associated with substance usage.	SubstanceDefinition	S3000L UoF Part Definition
substanceUsageCategory	ClassificationType	substanceUsageCategory is a classification which identifies possible limitations in substance usage.	SubstanceDefinition	S3000L UoF Part Definition
subSubSystem	umlString	subSubSystem is a string of characters that represents the sub-subsystem attribute of the data module code. **Note** A subSubSystem must be created in accordance with the rules defined in S1000D.	S1000DDataModule	S3000L UoF Document
subSystem	umlString	subSystem is a string of characters that represents the subsystem attribute of the data module code. **Note** A subSystem must be created in accordance with the rules defined in S1000D.	S1000DDataModule	S3000L UoF Document

Attribute Name	Type	Definition	Class Name	UoF
subtaskAcceptanceParameterDescription	DescriptorType	subtaskAcceptanceParameterDescription is a description that gives more information on the criteria that determines whether a Subtask is completed or not.	SubtaskAcceptanceParameter	S3000L UoF Task
subtaskAcceptanceParameterName	NameType	subtaskAcceptanceParameterName is a name by which the SubtaskAcceptanceParameter is known and can be easily referenced.	SubtaskAcceptanceParameter	S3000L UoF Task
subtaskAcceptanceParameterValue	PropertyType	subtaskAcceptanceParameterValue is a property that specifies the value (criteria) that must be fulfilled before the Subtask can be ended.	SubtaskAcceptanceParameter	S3000L UoF Task
subtaskAcceptanceParameterValueComparisonOperator	ClassificationType	subtaskAcceptanceParameterValueComparisonOperator is a classification that identifies the comparison operator against which a recorded value is to be evaluated against the subtaskAcceptanceParameterValue.	SubtaskAcceptanceParameter	S3000L UoF Task
subtaskDescription	DescriptorType	subtaskDescription is a description of the procedure performed during the Subtask.	SubtaskByDefinition	S3000L UoF Task
subtaskDuration	PropertyType	subtaskDuration is a property that specifies the average time required for the performance of a Subtask, regardless of the number of personnel working simultaneously. **Note** subtaskDuration does not include time spent awaiting spares, support equipment, facilities or personnel (logistics delay time).	SubtaskByDefinition	S3000L UoF Task
subtaskEndItemObjectiveState	StateType	subtaskEndItemObjectiveState is a state that identifies the condition of the Product that will exist after the accomplishment of the Subtask.	SubtaskByDefinition	S3000L UoF Task

Attribute Name	Type	Definition	Class Name	UoF
subtaskIdentifier	IdentifierType	subtaskIdentifier is an identifier that establishes a unique designator for a Subtask and to differentiate it from other instances of Subtask. **Note** A Subtask is identified within the context of a specific Task.	Subtask	S3000L UoF Task
subtaskInformationCode	ClassificationType	subtaskInformationCode is a classification that identifies the main purpose for the Subtask. **Note** A subtaskInformationCode must be created in accordance with the rules defined in S1000D.	SubtaskByDefinition	S3000L UoF Task
subtaskMaintenanceLocation	ClassificationType	subtaskMaintenanceLocation is a classification that specifies where the Subtask will be performed in relation to the Product. **Note** Proposed values equal the S1000D Item Location Codes.	SubtaskByDefinition	S3000L UoF Task
subtaskName	NameType	subtaskName is a name by which the Subtask is known and can be easily referenced.	SubtaskByDefinition	S3000L UoF Task
subtaskRole	ClassificationType	subtaskRole is a classification that identifies how the Subtask is related to the main function of the Task. **Note** subtaskRole enables mapping between S3000L and the main portions of the S1000D procedure schema.	Subtask	S3000L UoF Task
subtaskTimelineEvent	ClassificationType	subtaskTimelineEvent is a classification that identifies how the starting point for a Subtask depends upon its preceding Subtask.	SubtaskTimeline	S3000L UoF Task
subtaskTimelineLag	PropertyType	subtaskTimelineLag is a property that specifies the time that must elapse before the Subtask under consideration can start in relation to its associated subtaskTimelineEvent.	SubtaskTimeline	S3000L UoF Task

Attribute Name	Type	Definition	Class Name	UoF
system	umlString	system is a string of characters that represents the system attribute of the data module code. **Note** A system must be created in accordance with the rules defined in S1000D.	S1000DDataModule	S3000L UoF Document
systemDifferenceCode	umlString	systemDifferenceCode is a string of characters that represents the system difference code attribute of the data module code. **Note** A systemDifferenceCode must be created in accordance with the rules defined in S1000D.	S1000DDataModule	S3000L UoF Document
taskDuration	PropertyType	taskDuration is a property that specifies the average time required for the performance of a Task, regardless of the number of personnel working simultaneously. **Note** taskDuration does not include time spent awaiting spares, support equipment, facilities or personnel (logistics delay time). **Note** taskDuration could be calculated from the subtask durations.	TaskRevision	S3000L UoF Task
taskFrequencyCalculationMethod	DescriptorType	taskFrequencyCalculationMethod is a description that gives further information on how the TaskFrequencyValue has been derived.	TaskFrequency	S3000L UoF Task Usage
taskFrequencyValue	PropertyType	taskFrequencyValue is a property that represents the rate of occurrence value.	TaskFrequency	S3000L UoF Task Usage
taskIdentifier	IdentifierType	taskIdentifier is an identifier that establishes a unique designator for a Task and to differentiate it from other instances of Task.	Task	S3000L UoF Task
taskInformationCode	ClassificationType	taskInformationCode is a classification that identifies the main purpose for the Task. **Note** Valid classifications are defined in the S1000D.	TaskRevision	S3000L UoF Task

Attribute Name	Type	Definition	Class Name	UoF
taskInfrastructureResourceCategory	ClassificationType	taskInfrastructureResourceCategory is a classification that identifies further specialization for a TaskInfrastructureResource.	TaskInfrastructureResource	S3000L UoF Task Resource
taskInfrastructureResourceQuantity	PropertyType	taskInfrastructureResourceQuantity is a property that specifies the number of the TaskInfrastructureResource.	TaskInfrastructureResource	S3000L UoF Task Resource
taskMaterialResourceCategory	ClassificationType	taskMaterialResourceCategory is a classification that defines the role of the TaskMaterialResource in the context of the specified Task.	TaskMaterialResource	S3000L UoF Task Resource
taskMaterialResourceQuantity	PropertyType	taskMaterialResourceQuantity is a property that specifies the number of a TaskMaterialResource.	TaskMaterialResource	S3000L UoF Task Resource
taskName	NameType	taskName is a name by which the Task is known and can be easily referenced.	TaskRevision	S3000L UoF Task
taskOperabilityImpact	ClassificationType	taskOperabilityImpact is a classification that indicates the operational status and mission readiness of the end item during the Task.	TaskRevision	S3000L UoF Task
taskPersonnelResourceLaborTime	PropertyType	taskPersonnelResourceLaborTime is a property that specifies the time expended by the required TaskPersonnelResource. **Note** Labor time can be given as single values but also as ranges or as text eg, "AsRequired".	TaskPersonnelResource	S3000L UoF Task Resource
taskPersonnelResourceQuantity	PropertyType	taskPersonnelResourceQuantity is a property that specifies the number of the required TaskPersonnelResource **Note** Quantities can be given as single values but also as ranges or as text eg, "AsRequired".	TaskPersonnelResource	S3000L UoF Task Resource

Attribute Name	Type	Definition	Class Name	UoF
taskPersonnelResourceRole	ClassificationType	taskPersonnelResourceRole is a classification that defines the purpose of the required TaskPersonnelResource. **Example** – Man A – Man B – Performer – Quality assurance – Supervisor	TaskPersonnelResource	S3000L UoF Task Resource
taskPersonnelSafetyCriticality	ClassificationType	taskPersonnelSafetyCriticality is a classification that identifies the most serious health aspects that the performance of the Task can pose on personnel performing the Task.	TaskRevision	S3000L UoF Task
taskProductIntegrityCriticality	ClassificationType	taskProductIntegrityCriticality is a classification that identifies whether the Task is critical ie, if failure to accomplish it would result in adverse effects on product or system reliability, efficiency, effectiveness, safety, or cost. **Note** A task will also be designated as critical whenever system design characteristics approach human limitations, and thereby, significantly increase the likelihood of degraded, delayed, or otherwise impaired mission performance.	TaskRevision	S3000L UoF Task
taskRequirementAuthority	Organization	taskRequirementAuthority identifies the Organization that is the authoritative source for the identified TaskRequirement.	AuthorityDrivenTaskRequirement	S3000L UoF Task Requirement
taskRequirementAuthoritySourceType	ClassificationType	taskRequirementAuthoritySourceType is a classification that indicates the significance of the source from which TaskRequirement is derived.	AuthorityDrivenTaskRequirement	S3000L UoF Task Requirement
taskRequirementDescription	DescriptorType	taskRequirementDescription is a description that summarizes the procedure that needs to be performed based on the outcome of a support analysis activity.	TaskRequirementRevision	S3000L UoF Task Requirement

Attribute Name	Type	Definition	Class Name	UoF
taskRequirementIdentifier	IdentifierType	taskRequirementIdentifier is an identifier that establishes a unique designator for a TaskRequirement and to differentiate it from other instances of TaskRequirement.	TaskRequirement	S3000L UoF Task Requirement
taskRequirementInformationCode	ClassificationType	taskRequirementInformationCode is a classification that identifies the main purpose for the TaskRequirement. **Note** Valid classifications are defined in the S1000D.	TaskRequirementRevision	S3000L UoF Task Requirement
taskRequirementJustificationDescription	DescriptorType	taskRequirementJustificationDescription is a description that summarizes the need for a Task to meet the results from a support analysis activity.	TaskRequirementJustification	S3000L UoF Task Requirement
taskRequirementRevisionChangeDescription	DescriptorType	taskRequirementRevisionChangeDescription is a description that gives more information on content that has been altered between two revisions of a TaskRequirement.	TaskRequirementRevision	S3000L UoF Task Requirement
taskRequirementRevisionDate	DateType	taskRequirementRevisionDate is a date that specifies when a TaskRequirementRevision was defined.	TaskRequirementRevision	S3000L UoF Task Requirement
taskRequirementRevisionIdentifier	IdentifierType	taskRequirementRevisionIdentifier is an identifier that establishes a unique designator for a TaskRequirementRevision and to differentiate it from other instances of TaskRequirementRevision.	TaskRequirementRevision	S3000L UoF Task Requirement
taskRequirementRevisionRationale	DescriptorType	taskRequirementRevisionRationale is a description that gives more information on the justification for revising the TaskRequirement.	TaskRequirementRevision	S3000L UoF Task Requirement
taskRequirementRevisionStatus	StateType	taskRequirementRevisionStatus is a state that identifies the maturity of a TaskRequirementRevision.	TaskRequirementRevision	S3000L UoF Task Requirement
taskRequirementSpecialResourceRequirement	DescriptorType	taskRequirementSpecialResourceRequirement is a description that gives more information on unusual resources which are needed for the performance of the required Task.	TaskRequirementRevision	S3000L UoF Task Requirement

Attribute Name	Type	Definition	Class Name	UoF
taskResourceDuration	PropertyType	taskResourceDuration is a property that specifies the average time that a TaskResource is needed to perform a specified amount of work.	TaskResource	S3000L UoF Task Resource
taskResourceFixedResourceMarker	umlBoolean	taskResourceFixedResourceMarker is a boolean that identifies that no resource other than the one specified is allowed to be used as the TaskResourceItem.	TaskResource	S3000L UoF Task Resource
taskResourceIdentifier	IdentifierType	taskResourceIdentifier is an identifier that establishes a unique designator for a TaskResource and to differentiate it from other instances of TaskResource.	TaskResource	S3000L UoF Task Resource
taskResourceRelationshipType	ClassificationType	taskResourceRelationshipType is a classification that identifies the meaning of the established relationship.	TaskResourceRelationship	S3000L UoF Task Resource
taskRevisionChangeDescription	DescriptorType	taskRevisionChangeDescription is a description that gives more information on content that has been altered between two revisions of a Task.	TaskRevision	S3000L UoF Task
taskRevisionDate	DateType	taskRevisionDate is a date that specifies when the Task was revised.	TaskRevision	S3000L UoF Task
taskRevisionIdentifier	IdentifierType	taskRevisionIdentifier is an identifier that establishes a unique designator for a TaskRevision and to differentiate it from other instances of TaskRevision.	TaskRevision	S3000L UoF Task
taskRevisionRationale	DescriptorType	taskRevisionRationale is a description that gives more information on the justification for revising the Task.	TaskRevision	S3000L UoF Task
taskRevisionStatus	StateType	taskRevisionStatus is a state that identifies the progress on the development of a TaskRevision.	TaskRevision	S3000L UoF Task

Attribute Name	Type	Definition	Class Name	UoF
taskTotalLaborTime	PropertyType	taskTotalLaborTime is a property that specifies the total time to be expended during a Task. **Note** taskTotalLaborTime includes the labor time for all required task personnel resources.	TaskRevision	S3000L UoF Task
technologyBehaviorKnowledgeRating	DatedClassification	technologyBehaviorKnowledgeRating is a classification that identifies to which extent knowledge exist on the behavior of the technology used, during normal use or caused by special event.	TechnologyBehaviorRating	S3000L UoF Damage Definition
technologySensitivityRating	DatedClassification	technologySensitivityRating is a classification that identifies the likelihood that the AnalysisCandidateItem will be subject for damage during its life cycle.	TechnologyBehaviorRating	S3000L UoF Damage Definition
thresholdValue	PropertyType	thresholdValue is a property that represents the value that is measured and evaluated as part of the ParameterThresholdDefinition.	ParameterThresholdDefinition	S3000L UoF Time Limit
thresholdValueQualifier	ClassificationType	thresholdValueQualifier is a classification that specifies a constraint to be used when evaluating an actual value against the thresholdValue.	ParameterThresholdDefinition	S3000L UoF Time Limit
timeLimitDescription	DescriptorType	timeLimitDescription is a description that provides a human readable expression of the defined time limit expression.	TimeLimit	S3000L UoF Time Limit
timeLimitHarmonizationIndicator	umlBoolean	timeLimitHarmonizationIndicator is a boolean that identifies if the TimeLimit is the result from a Task packaging procedure.	TimeLimit	S3000L UoF Time Limit
tradeName	NameType	tradeName is a name that uniquely establishes a craft or profession.	Trade	S3000L UoF Competence Definition
warningCautionNoteDescription	DescriptorType	warningCautionNoteDescription is a description that gives more information on safety, legal and health considerations.	WarningCautionNote	S3000L UoF Task

Attribute Name	Type	Definition	Class Name	UoF
warningCautionNoteIdentifier	IdentifierType	warningCautionNoteIdentifier is an identifier that establishes a unique designator for a warning, caution or note, and to differentiate it from other instances of warning , caution or note.	WarningCautionNote	S3000L UoF Task
warningCautionNoteType	ClassificationType	warningCautionNoteType is a classification that identifies severity and scope for the safety, legal and health considerations.	WarningCautionNote	S3000L UoF Task
zoneElementType	ClassificationType	zoneElementType is a classification that identifies further specialization for a ZoneElement.	ZoneElement	S3000L UoF Zone Element

4 Valid values

The full list of S3000L attribute valid values is provided in Table 4.

Note

The specification does not yet define valid values for all classification types.

Note

Specified valid values are recommended values and projects can therefore tailor the valid values as needed (refer to SX005G).

Table 4 List of valid values

Attribute name	Valid value	Valid value name
aggregatedElementType	FA	SX001G:familyBreakdownElement
	FU	SX001G:functionBreakdownElement
	GR	SX001G:groupBreakdownElement
	SY	SX001G:systemBreakdownElement
allowedProductConfigurationIdentifier	ID	SX001G:allowedProductConfigurationIdentifier
analysisActivityDecision	O	SX001G:toBeDecidedAnalysisActivity
	R	SX001G:rejectedAnalysisActivity
	S	SX001G:selectedAnalysisActivity
analysisActivityRevisionIdentifier	ID	SX001G:analysisActivityRevisionIdentifier
analysisActivityRevisionStatus		Refer to revisionStatusCode valid value library, Para 5.
analysisActivityType	CMP	SX001G:lsaComparativeAnalysis
	COR	SX001G:lsaCorrectiveMaintenanceAnalysis
	DMG	SX001G:lsaDamageAnalysis
	HF	SX001G:lsaHumanFactorAnalysis
	LORA	SX001G:lsaLevelOfRepairAnalysis
	MNT	SX001G:lsaMaintainabilityAnalysis
	MTA	SX001G:lsaMaintenanceTaskAnalysis
	OP	SX001G:lsaOperationalAnalysis
	OTH	SX001G:lsaOtherAnalysis
	REL	SX001G:lsaReliabilityAnalysis
	SEV	SX001G:lsaSpecialEventAnalysis
	SIM	SX001G:lsaSimulationOperationalScenariosAnalysis
	SWD	SX001G:lsaSoftwareDataLoadingAnalysis

Attribute name	Valid value	Valid value name
	SWS	SX001G:lsaSoftwareSupportAnalysis
	TNA	SX001G:lsaTrainingNeedsAnalysis
	TST	SX001G:lsaTestabilityAnalysis
analysisCandidateItemSelectionIndicator	F	SX001G:fullCandidateItem
	N	SX001G:nonCandidateItem
	O	SX001G:toBeDecidedCandidateItem
	P	SX001G:partialCandidateItem
applicabilityStatementIdentifier	ID	SX001G:applicabilityStatementIdentifier
authorityToOperateIdentifier	ID	SX001G:authorityToOperateIdentifier
breakdownElementEssentiality	1	SX001G:criticalBreakdownElement
	2	SX001G:partialCriticalBreakdownElement
	3	SX001G:nonCriticalBreakdownElement
breakdownElementIdentifier	ASD	SX001G:asdSystemHardwareIdentificationCode
	CSN	SX001G:figureItemIdentifier
	ID	SX001G:breakdownElementIdentifier
	LCN	SX001G:fullLogisticsSupportAnalysisControlNumber
	SNS	SX001G:standardNumberingSystemIdentifier
breakdownElementRevisionIdentifier	ID	SX001G:breakdownElementRevisionIdentifier
breakdownElementRevisionRelationshipType	ALT	SX001G:alternateToBreakdownElement
	AP	SX001G:breakdownElementAccessPoint
	FP	SX001G:functionalToPhysicalBreakdownElementRelationship
breakdownElementRevisionStatus		Refer to revisionStatusCode valid value library, Para 5.
breakdownElementUsageIdentifier	ID	SX001G:breakdownElementUsageIdentifier
breakdownElementUsageRelationshipType	AND	SX001G:mutualBreakdownElementInclusion
	XOR	SX001G:mutualBreakdownElementExclusion
breakdownRevisionIdentifier	ID	SX001G:breakdownRevisionIdentifier
breakdownRevisionRelationshipType	BO	SX001G:basedOnBreakdownRevision
	M	SX001G:matchesBreakdownRevision
breakdownRevisionStatus		Refer to revisionStatusCode valid value library, Para 5.

Attribute name	Valid value	Valid value name
breakdownType	ASD	SX001G:asdSystemHardwareBreakdown
	FAM	SX001G:familyBreakdown
	FU	SX001G:functionalBreakdown
	PH	SX001G:physicalBreakdown
	PR	SX001G:provisioningBreakdown
	SY	SX001G:systemBreakdown
	ZONE	SX001G:zonalBreakdown
changeAuthorizationIdentifier	AMN	SX001G:changeAmendmentNumber
	CAN	SX001G:changeAuthorizationNumber
	ID	SX001G:changeAuthorizationIdentifier
changeNotificationType	A	SX001G:applicabilityChangeNotification
	E	SX001G:editorialChangeNotification
	M	SX001G:markupChangeNotification
	T	SX001G:technicalChangeNotification
changeRequestIdentifier	ID	SX001G:changeRequestIdentifier
changeRequestStatus	A	SX001G:approvedChangeRequest
	IW	SX001G:inWorkChangeRequest
	R	SX001G:rejectedChangeRequest
	S	SX001G:submittedChangeRequest
circuitBreakerIdentifier	ID	SX001G:circuitBreakerIdentifier
circuitBreakerSettingIdentifier	ID	SX001G:circuitBreakerSettingIdentifier
circuitBreakerSettingsIdentifier	ID	SX001G:circuitBreakerSettingsIdentifier
circuitBreakerState	C	SX001G:closeCircuitBreakerState
	O	SX001G:openCircuitBreakerState
	VC	SX001G:verifyCloseCircuitBreakerState
	VO	SX001G:verifyOpenCircuitBreakerState
circuitBreakerType	CLIP	SX001G:dummyCircuitBreaker
	ELMEC	SX001G:electroMechanicCircuitBreaker
	ELTRO	SX001G:electronicCircuitBreaker
competencyDefinitionIdentifier	ID	SX001G:competencyDefinitionIdentifier

Attribute name	Valid value	Valid value name
competencyDefinitionType	CRED	SX001G:credentialCompetencyDefinition
	EXPE	SX001G:experienceCompetencyDefinition
	JOBT	SX001G:jobTitleCompetencyDefinition
conditionInstanceIdentifier	ID	SX001G:conditionInstanceIdentifier
	SB	SX001G:serviceBulletinIdentifier
conditionTypeAssertMemberAssertValueComparisonOperator		Refer to comparisonOperatorCode valid value library, Para 5.
contractIdentifier	ID	SX001G:contractIdentifier
contractPartyRole	CTR	SX001G:contractor
	CUS	SX001G:customer
	SUB	SX001G:subContractor
	USER	SX001G:user
contractRelationshipType	EXTC	SX001G:extendsContract
	RELC	SX001G:relatedContract
	REPC	SX001G:replacesContract
	SUBC	SX001G:subContractOf
countryCode		Refer to countryCodeValues valid value library, Para 5.
damageDefinitionFamily	CRK	SX001G:crackDamageDefinition
	DNT	SX001G:dentDamageDefinition
	SCR	SX001G:scratchDamageDefinition
dataModuleIssueLanguage		Refer to languageCodeValues valid value library, Para 5.
dataModuleIssueLanguageCountry		Refer to countryCodeValues valid value library, Para 5.
decisionTreeTemplateActionDefinitionIdentifier	ID	SX001G:decisionTreeTemplateActionDefinitionIdentifier
decisionTreeTemplateIdentifier	ID	SX001G:decisionTreeTemplateIdentifier
decisionTreeTemplateQuestionDefinitionIdentifier	ID	SX001G:decisionTreeTemplateQuestionDefinitionIdentifier
decisionTreeTemplateRevisionIdentifier	ID	SX001G:decisionTreeTemplateRevisionIdentifier
decisionTreeTemplateRevisionStatus		Refer to revisionStatusCode valid value library, Para 5.
detectionMeansAlarmIdentifier	ID	SX001G:detectionMeanAlarmIdentifier

Attribute name	Valid value	Valid value name
detectionMeansAlarmType	BIT	SX001G:builtInTest
	CBIT	SX001G:continuousBuiltInTest
	GSE	SX001G:groundSupportEquipmentTest
	IBIT	SX001G:initiatedBuiltInTest
	PBIT	SX001G:powerOnBuiltInTest
digitalFileContentClass	3D	SX001G:3dModelFileContent
	AUD	SX001G:audioFileContent
	DRW	SX001G:drawingFileContent
	MOV	SX001G:movieFileContent
	PHO	SX001G:photographFileContent
	PUB	SX001G:publicationFileContent
	REP	SX001G:reportFileContent
	SCH	SX001G:schematicsFileContent
	WIR	SX001G:wiringFileContent
digitalFileLocator	ID	SX001G:digitalFileLocator
digitalFileType	ASF	SX001G:advanceSystemsFormatFileType
	AVI	SX001G:audioVideoInterleavedFileType
	BIN	SX001G:binaryFileType
	CGM	SX001G:computerGraphicsMetafileFileType
	DAT	SX001G:dataFormatFileType
	DOC	SX001G:microsoftWordFormatFileType
	DOCX	SX001G:microsoftWordFormatFileType
	EDF	SX001G:europeanDataFormatSignalFileType
	HTM	SX001G:hyperTextMarkupLanguageFileType
	JPEG	SX001G:jointPhotographicExpertsGroupFileType
	MOV	SX001G:quickTimeMovieFileType
	MP3	SX001G:mp3AudioFileType
	MPEG	SX001G:motionPictureExpertsGroupMovieFileType
	ODS	SX001G:openDocumentSpreadsheetFileType
	ODT	SX001G:openDocumentTextFileType
	PDF	SX001G:portableDocumentFormatFileType
	PNG	SX001G:portableNetworkGraphicsFileType

Attribute name	Valid value	Valid value name
	RAW	SX001G:rawSampleAudioFileType
	TIFF	SX001G:taggedImageFileFormatFileType
	TXT	SX001G:textFileType
	UNK	SX001G:unknownFileType
	WAV	SX001G:waveformAudioFileType
	XLS	SX001G:microsoftExcelFormatFileType
	XLSX	SX001G:microsoftExcelFormatFileType
	XSD	SX001G:xmlSchemaDefinitionFileType
documentIdentifier	ID	SX001G:documentIdentifier
documentIssueIdentifier	ID	SX001G:documentIssueIdentifier
documentType	DRW	SX001G:drawingDocument
	PCAT	SX001G:partsCatalogueDocument
	SPEC	SX001G:specificationDocument
	STD	SX001G:standardsDocument
	TMAN	SX001G:technicalManual
	TR	SX001G:technicalReport
environmentDefinitionCharacteristicValueComparisonOperator		Refer to comparisonOperatorCode valid value library, Para 5.
environmentDefinitionIdentifier	ID	SX001G:environmentDefinitionIdentifier
environmentDefinitionRelationshipType	SPEC	SX001G:specializationOfEnvironmentDefinition
	SSNL	SX001G:seasonalEnvironmentDefinitionVariation
environmentDefinitionRevisionIdentifier	ID	SX001G:environmentDefinitionRevisionIdentifier
environmentDefinitionRevisionStatus		Refer to revisionStatusCode valid value library, Para 5.
evaluationByAssertionRole	CUS	SX001G:customer
	OP	SX001G:operator
facilityIdentifier	ID	SX001G:facilityIdentifier
facilityRelationshipType	INCL	SX001G:includesFacility
failureModeAnalysisRevisionIdentifier	ID	SX001G:failureModeAnalysisRevisionIdentifier
failureModeAnalysisRevisionStatus		Refer to revisionStatusCode valid value library, Para 5.

Attribute name	Valid value	Valid value name
failureModeAnalysisType	E	SX001G:equipmentFailureModeAnalysis
	F	SX001G:systemFunctionFailureModeAnalysis
failureModeCauseIdentifier	ID	SX001G:failureModeCauseIdentifier
failureModeCauseItemRelationshipType	A	SX001G:failureCauseAffectedItem
	C	SX001G:failureCauseConsequenceOf
failureModeCriticality	I	SX001G:catastrophicFailureModeConsequence
	II	SX001G:criticalFailureModeConsequence
	III	SX001G:marginalFailureModeConsequence
	IV	SX001G:negligibleFailureModeConsequence
failureModeEffectLevel	E	SX001G:systemEndItemFailureModeEffect
	H	SX001G:higherFailureModeEffect
	L	SX001G:localFailureModeEffect
	N	SX001G:nextHigherFailureModeEffect
failureModeIdentifier	ID	SX001G:failureModeIdentifier
failureModeIsolationAbilityRating	H	SX001G:highLikelihoodFailureModeIsolationRating
	L	SX001G:lowLikelihoodFailureModeIsolationRating
	M	SX001G:mediumLikelihoodFailureModeIsolationRating
	MH	SX001G:moderatelyHighLikelihoodFailureModeIsolationRating
	ML	SX001G:moderatelyLowLikelihoodFailureModeIsolationRating
failureModeSymptomsSignatureIdentifier	ID	SX001G:failureModeSymptomsSignatureIdentifier
geographicalAreaType	ADM	SX001G:administrativeRegion
	CITY	SX001G:city
	CON	SX001G:continent
	CTYG	SX001G:countryGroup
	DES	SX001G:desert
	MUL	SX001G:multiGeographicalArea
	OCE	SX001G:ocean
	REG	SX001G:geographicalRegion
	SEA	SX001G:sea

Attribute name	Valid value	Valid value name
geographicalCoordinateSystem	DD	SX001G:decimalDegreeGeographicalCoordinateSystem
	DMS	SX001G:degreeMinutesSecondsGeographicalCoordinateSystem
hardwareElementRepairability	N	SX001G:nonRepairableHardwareElement
	P	SX001G:partialRepairableHardwareElement
	R	SX001G:repairableHardwareElement
hardwareElementRepairabilityStrategy	B	SX001G:reconditionHardwareElement
	D	SX001G:limitedRepairAtLowerLevelOverhaulAtDepotLevelHardwareElement
	F	SX001G:repairAtIntermediateLevelHardwareElement
	L	SX001G:repairAtIndustryLevelHardwareElement
	O	SX001G:repairAtOrganizationalLevelHardwareElement
	Z	SX001G:nonRepairableHardwareElement
hardwareElementReplaceability	N	SX001G:nonReplaceableHardwareElement
	R	SX001G:replaceableHardwareElement
hardwareElementReplaceabilityStrategy	LINE	SX001G:lineReplaceableHardwareElement
	SHOP	SX001G:shopReplaceableHardwareElement
hardwareElementStructuralIndicator	SD	SX001G:structuralDetail
	SI	SX001G:structuralItem
	SSI	SX001G:structuralSignificantItem
hardwareElementType	EQP	SX001G:equipmentHardwareElement
	OPN	SX001G:openingHardwareElement
	PNL	SX001G:panelHardwareElement
hardwarePartEnvironmentalAspectInUseClass	A	SX001G:acidificationPart
	B	SX001G:energyRegainedByBurningPart
	G	SX001G:greenhouseEffectPart
	H	SX001G:harmfulToEnvironmentPart
	O	SX001G:dangerousForOzoneLayerPart
	R	SX001G:materialRecyclingPart
	W	SX001G:materialWastePart
hardwarePartEnvironmentalAspectPlannedDisposalClass	A	SX001G:acidificationPart
	B	SX001G:energyRegainedByBurningPart

Attribute name	Valid value	Valid value name
	G	SX001G:greenhouseEffectPart
	H	SX001G:harmfulToEnvironmentPart
	O	SX001G:dangerousForOzoneLayerPart
	R	SX001G:materialRecyclingPart
	W	SX001G:materialWastePart
hardwarePartFitmentRequirement	MAJ	SX001G:majorFitmentRequiredPart
	MIN	SX001G:minorFitmentRequiredPart
	NO	SX001G:noFitmentRequiredPart
hardwarePartHazardousClass		Refer to hardwarePartHazardousClassCode valid value library, Para 5.
hardwarePartLogisticsCategory	C	SX001G:consumablePart
	E	SX001G:expendableSparePart
	EP	SX001G:expendablePersonalProtectionPart
	HT	SX001G:standardHandTool
	IT	SX001G:informationTechnologyPart
	M	SX001G:rawMaterial
	PA	SX001G:packagingPart
	PE	SX001G:personalProtectionPart
	PR	SX001G:productProtectionPart
	R	SX001G:repairableSparePart
	S	SX001G:sparePart
	SE	SX001G:supportEquipment
hardwarePartMaintenanceSchedulingStart	A	SX001G:maintenanceSchedulingStartAtInstallationInAssemblyPart
	D	SX001G:maintenanceSchedulingStartsAtDeliveryPart
	E	SX001G:maintenanceSchedulingStartsAtInstallationInEndItemPart
	P	SX001G:maintenanceSchedulingStartsAtProductionPart
hardwarePartRepairability	N	SX001G:nonRepairablePart
	P	SX001G:partialRepairablePart
	R	SX001G:repairablePart
hardwarePartRepairabilityStra	B	SX001G:reconditionPart

Attribute name	Valid value	Valid value name
tegy	D	SX001G:limitedRepairAtLowerLevelOverhaulAtDepotLevelPart
	F	SX001G:repairAtIntermediateLevelPart
	L	SX001G:repairAtIndustryLevelPart
	O	SX001G:repairAtOrganizationalLevelPart
	Z	SX001G:nonRepairablePart
hardwarePartSupportFamilyClass	GBX	SX001G:gearBoxPartSupportFamilyClass
	HTR	SX001G:heaterPartSupportFamilyClass
	PMP	SX001G:pumpPartSupportFamilyClass
	VLV	SX001G:valvePartSupportFamilyClass
hardwarePartTraceabilityClass	B	SX001G:batchTraceabilityPart
	N	SX001G:noTraceabilityRequiredPart
	S	SX001G:serializedTraceabilityPart
infrastructureComplianceLevel	F	SX001G:fullInfrastructureCompliance
	N	SX001G:nonInfrastructureCompliance
	P	SX001G:partialInfrastructureCompliance
inServiceOptimizationAnalysisRevisionIdentifier	ID	SX001G:inServiceOptimizationAnalysisRevisionIdentifier
inServiceOptimizationAnalysisRevisionStatus		Refer to revisionStatusCode valid value library, Para 5.
inServiceOptimizationAnalysisStepIdentifier	ID	SX001G:inServiceOptimizationAnalysisStepIdentifier
locationRelationshipType	IN	SX001G:locationLocatedIn
	NXT	SX001G:locationLocatedNextTo
lsaFailureModeGroupIdentifier	ID	SX001G:lsaFailureModeGroupIdentifier
lsaFailureModeGroupTaskRequirementPurpose	C	SX001G:failureModeCorrectiveTaskRequirement
	I	SX001G:failureModeIsolationTaskRequirement
	T	SX001G:failureModeTestTaskRequirement
maintenanceFacilityType	BAT	SX001G:batteryWorkshop
	CAL	SX001G:calibrationWorkshop
	HAN	SX001G:hangar
	HYD	SX001G:hydraulicWorkshop
	REP	SX001G:repairWorkshop
	SHOP	SX001G:generalPurposeWorkshop

Attribute name	Valid value	Valid value name
maintenanceLevelIdentifier	ID	SX001G:maintenanceLevelIdentifier
maintenanceSignificantOrRelevant	N	SX001G:nonMaintenanceSignificantOrRelevantBreakdownElement
	R	SX001G:maintenanceRelevantBreakdownElement
	S	SX001G:maintenanceSignificantBreakdownElement
measurementPointDefinitionIdentifier	ID	SX001G:measurementPointDefiinitionIdentifier
measurementPointDefinitionType	G	SX001G:gaugeMeasurementPoint
measurementPointDegradedStateDefinitionAssertValueComparisonOperator		Refer to comparisonOperatorCode valid value library, Para 5.
measurementPointFaultStateDefinitionAssertValueComparisonOperator		Refer to comparisonOperatorCode valid value library, Para 5.
messageContentStatus	D	SX001G:draftMessageContent
	F	SX001G:finalMessageContent
	P	SX001G:preliminaryMessageContent
messageContentType	B	SX001G:baselineMessage
	U	SX001G:netChangeMessage
messageIdentifier	ID	SX001G:messageIdentifier
messageLanguage		Refer to languageCodeValues valid value library, Para 5.
messagePartyType	F	SX001G:messageForwarder
	R	SX001G:messageReceiver
	S	SX001G:messageSender
messageRelationshipType	A	SX001G:acknowledgementOfMessage
	O	SX001G:observationOnMessage
	R	SX001G:replyToMessage
	U	SX001G:updateToMessage
nonConformanceType	C	SX001G:concession
	W	SX001G:waiver
operatingBaseType	D	SX001G:deploymentOperatingBase
	F	SX001G:forwardOperatingBase
	M	SX001G:mainOperatingBase

Attribute name	Valid value	Valid value name
operatingLocationTypeIdentifier	ID	SX001G:operatingLocationTypeIdentifier
partAsDesignedPartsListRelationshipType	C	SX001G:correspondsToPartsList
	D	SX001G:derivedFromPartsList
	E	SX001G:extendsPartsList
partDemilitarizationClass	A	SX001G:demilitarizationNotRequiredPart
	B	SX001G:tradeSecurityControlAtDisposalPart
	C	SX001G:keyPointsDemilitarizationPart
	D	SX001G:mutilationDemilitarizationPart
	E	SX001G:nationalFurnishedDemilitarizationPart
	F	SX001G:managerFurnishedDemilitarizationPart
	G	SX001G:demilitarizationPriorToTransferPart
	R	SX001G:specificInstructionsDemilitarizationPart
	Y	SX001G:specificInstructionsDemilitarizationCryptoMaterial
partIdentifier	ID	SX001G:partIdentifier
	OEM	SX001G:originalEquipmentManufacturerPartNumber
	REF	SX001G:partReferenceNumber
	STD	SX001G:standardsReferenceDesignator
	SUP	SX001G:supplierPartNumber
partMaturityClass	COTS	SX001G:commonOffTheShelfPart
	CUST	SX001G:customerFurnishedPart
	MAM	SX001G:majorModificationOfExistingPart
	MOM	SX001G:moderateModificationOfExistingPart
	NEW	SX001G:newDevelopedPart
	OBS	SX001G:obsoletePart
partsListEntryIdentifier	ID	SX001G:partsListEntryIdentifier
	LN	SX001G:partsListLineNumber
partsListRevisionIdentifier	ID	SX001G:partsListRevisionIdentifier
partsListRevisionStatus		Refer to revisionStatusCode valid value library, Para 5.
partsListType	EBOM	SX001G:engineeringPartsList
	MBOM	SX001G:manufacturingPartsList

Attribute name	Valid value	Valid value name
	PBOM	SX001G:provisioningPartsList
	SBOM	SX001G:supportPartsList
performanceParameterRevisionIdentifier	ID	SX001G:performanceParameterRevisionIdentifier
performanceParameterRevisionStatus		Refer to revisionStatusCode valid value library, Para 5.
performanceParameterType	DMC	SX001G:directMaintenanceCostPerformanceParameter
	FOH	SX001G:failuresPerOperatingHourPerformanceParameter
	FR	SX001G:failureRatePerformanceParameter
	LCC	SX001G:lifeCycleCostPerformanceParameter
	MDT	SX001G:meanDownTimePerformanceParameter
	MFOP	SX001G:maintenanceFreeOperatingPeriodPerformanceParameter
	MMHOH	SX001G:maintenanceManHoursPerOperatingHourPerformanceParameter
	MTBF	SX001G:meanTimeBetweenFailurePerformanceParameter
	MTBUR	SX001G:meanTimeBetweenUnscheduledRemovalPerformanceParameter
	MTTR	SX001G:meanTimeToRepairPerformanceParameter
	PSL	SX001G:productServiceLifePerformanceParameter
	RT	SX001G:replacementTimePerformanceParameter
	SMI	SX001G:scheduledMaintenanceIntervalPerformanceParameter
	SPT	SX001G:shopProcessingTimePerformanceParameter
performanceParameterValueLimitQualifier	MAX	SX001G:maximumValueLimitQualifier
	MIN	SX001G:minimumValueLimitQualifier
physicalReplaceability	NOT	SX001G:nonReplaceablePart
	REPL	SX001G:replaceablePart
physicalReplaceabilityStrategy	LINE	SX001G:lineReplaceablePart
	SHOP	SX001G:shopReplaceablePart
productIdentifier	EIAC	SX001G:endItemAcronymCode
	ID	SX001G:productIdentifier
	MOI	SX001G:modelIdentificationCode

Attribute name	Valid value	Valid value name
productUsagePhaseRelationshipType	A	SX001G:happensAfterProductUsagePhase
	B	SX001G:happensBeforeProductUsagePhase
	D	SX001G:happensDuringProductUsagePhase
	P	SX001G:partOfProductUsagePhase
productVariantIdentifier	ID	SX001G:productVariantIdentifier
	MOI	SX001G:modelIdentificationCode
	UOC	SX001G:usableOnCode
projectIdentifier	ID	SX001G:projectIdentifier
	MOI	SX001G:modelIdentificationCode
proposedTrainingMethod	CR	SX001G:classroomTrainingMethod
	CW	SX001G:coursewareTrainingMethod
	OJT	SX001G:onTheJobTrainingMethod
publicationModuleIssueLanguage		Refer to languageCodeValues valid value library, Para 5.
publicationModuleIssueLanguageCountry		Refer to countryCodeValues valid value library, Para 5.
referencedDocumentRole	DES	SX001G:designDocumentReference
	DIR	SX001G:directiveDocumentReference
	DRW	SX001G:drawingDocumentReference
	REF	SX001G:generalDocumentReference
	REQ	SX001G:requirementsDocumentReference
	RES	SX001G:resultDocumentReference
	SRC	SX001G:sourceDocumentReference
	VAL	SX001G:validationDocumentReference
	VER	SX001G:verificationDocumentReference
referenceDesignator	RFD	SX001G:referenceDesignator
referencedOrganizationRole	APPR	SX001G:approverOrganization
	AUTH	SX001G:authorizingOrganization
	DSGN	SX001G:responsibleDesignOrganization
	PUBL	SX001G:publishedByOrganization
	REVW	SX001G:reviewOrganization
remarkType	INT	SX001G:internalRemark
	PUB	SX001G:publicRemark

Attribute name	Valid value	Valid value name
	RSP	SX001G:responseToRemark
resourceSpecificationIdentifier	ID	SX001G:resourceSpecificationIdentifier
	STD	SX001G:standardsReferenceDesignator
resourceSpecificationRevisionIdentifier	ID	SX001G:resourceSpecificationRevisionIdentifier
resourceSpecificationRevisionStatus		Refer to revisionStatusCode valid value library, Para 5.
resourceSpecificationType	C	SX001G:consumablePart
	CNWK	SX001G:communicationNetworkInfrastructureResource
	COMP	SX001G:computerInfrastructureResource
	DOCK	SX001G:dockInfrastructureResource
	DRYD	SX001G:dryDockInfrastructureResource
	E	SX001G:expendableSparePart
	EP	SX001G:expendablePersonalProtectionPart
	GAR	SX001G:garageInfrastructureResource
	HNG	SX001G:hangarInfrastructureResource
	HT	SX001G:standardHandTool
	INF	SX001G:infrastructureResource
	IT	SX001G:informationTechnologyPart
	M	SX001G:rawMaterial
	PA	SX001G:packagingPart
	PE	SX001G:personalProtectionPart
	POW	SX001G:powerInfrastructureResource
	PR	SX001G:productProtectionPart
	R	SX001G:repairableSparePart
	S	SX001G:sparePart
	SE	SX001G:supportEquipment
	SSE	SX001G:safetyRelatedSupportEquipment
	TNWK	SX001G:transportNetworkInfrastructureResource
skillCode	ID	SX001G:skillCode
softwareElementLoadingStrategy	IND	SX001G:industryLoadableSoftwareElement
	LINE	SX001G:lineLoadableSoftwareElement

Attribute name	Valid value	Valid value name
	SHOP	SX001G:shopLoadableSoftwareElement
softwareElementType	D	SX001G:distributedSoftwareElement
	E	SX001G:embeddedSoftwareElement
	L	SX001G:loadableSoftwareElement
softwarePartType	D	SX001G:distributedSoftwarePart
	E	SX001G:embeddedSoftwarePart
	L	SX001G:loadableSoftwarePart
specialEventDefinitionCauseCategory	ANI	SX001G:animalSpecialEvent
	CBT	SX001G:combatSpecialEvent
	DYS	SX001G:internalDysfunctionSpecialEvent
	EXT	SX001G:externalCauseSpecialEvent
	HEA	SX001G:extensiveHeatSpecialEvent
	HUM	SX001G:humanImpactSpecialEvent
	INT	SX001G:internalSpecialEvent
	MAN	SX001G:materialManeuverSpecialEvent
	MET	SX001G:meteorologicalSpecialEvent
	NAT	SX001G:naturalPhenomenonSpecialEvent
	VIB	SX001G:extensiveVibrationSpecialEvent
substanceIdentifier	CAS	SX001G:chemicalAbstractsServiceRegistryNumber
	ID	SX001G:substanceIdentifier
substanceRiskFactor	1	SX001G:minimalHealthOrPhysicalHazardSubstance
	2	SX001G:lowHealthOrPhysicalHazardSubstance
	3	SX001G:moderateHealthOrPhysicalHazardSubstance
	4	SX001G:highHealthOrPhysicalHazardSubstance
substanceUsageCategory	F	SX001G:forbiddenSubstance
	L	SX001G:authorizedWithLimitationSubstance
	N	SX001G:authorizedWithoutLimitationSubstance
subtaskAcceptanceParameterValueComparisonOperator		Refer to comparisonOperatorCode valid value library, Para 5.
subtaskEndItemObjectiveState	AS	SX001G:airSupplyEstablishedEndItemState
	CS	SX001G:controlStatusEstablishedEndItemState
	DF	SX001G:defueledEndItemState

Attribute name	Valid value	Valid value name
	EL	SX001G:electricalPowerEstablishedEndItemState
	ELA	SX001G:electricalPowerFromAPUEstablishedEndItemState
	ELE	SX001G:electricalPowerFromEngineEstablishedEndItemState
	ELI	SX001G:internalElectricalPowerEstablishedEndItemState
	ELX	SX001G:externalElectricalPowerEstablishedEndItemState
	FU	SX001G:fueledEndItemState
	HP	SX001G:hydraulicPowerEstablishedEndItemState
	JC	SX001G:jackedEndItemState
	SY	SX001G:safetyDeviceEstablishedEndItemState
	TC	SX001G:taskCheckedEndItemState
	UJ	SX001G:unjackedEndItemState
	WS	SX001G:waterSupplyEstablishedEndItemState
subtaskIdentifier	ID	SX001G:subtaskIdentifier
subtaskInformationCode		Refer to taskInformationCodeValues valid value library, Para 5.
subtaskMaintenanceLocation	A	SX001G:maintenanceOnItemWhenInstalledOnProduct
	B	SX001G:maintenanceOnItemOnMajorAssembly
	C	SX001G:maintenanceOnBench
	D	SX001G:maintenanceAnywhere
subtaskRole	CL	SX001G:closeupSubtask
	CON	SX001G:coreNoRequiredConditionsSubtask
	COR	SX001G:coreSubtask
	ST	SX001G:startupSubtask
subtaskTimelineEvent	END	SX001G:subtaskEndEvent
	START	SX001G:subtaskStartEvent
taskIdentifier	ID	SX001G:taskIdentifier
taskInformationCode		Refer to taskInformationCodeValues valid value library, Para 5.
taskInfrastructureResourceCategory	CNWK	SX001G:communicationNetworkInfrastructureResource
	COMP	SX001G:computerInfrastructureResource

Attribute name	Valid value	Valid value name
	DOCK	SX001G:dockInfrastructureResource
	DRYD	SX001G:dryDockInfrastructureResource
	GAR	SX001G:garageInfrastructureResource
	HNG	SX001G:hangarInfrastructureResource
	POW	SX001G:powerInfrastructureResource
	TNWK	SX001G:transportNetworkInfrastructureResource
taskMaterialResourceCategory	C	SX001G:consumablePart
	E	SX001G:expendableSparePart
	EP	SX001G:expendablePersonalProtectionPart
	HT	SX001G:standardHandTool
	IT	SX001G:informationTechnologyPart
	M	SX001G:rawMaterial
	PA	SX001G:packagingPart
	PE	SX001G:personalProtectionPart
	PR	SX001G:productProtectionPart
	R	SX001G:repairableSparePart
	S	SX001G:sparePart
	SE	SX001G:supportEquipment
	SSE	SX001G:safetyRelatedSupportEquipment
taskOperabilityImpact	A	SX001G:systemInoperableDuringTaskExecution
	B	SX001G:systemOperableDuringTaskExecution
	C	SX001G:systemFullMissionCapableDuringTaskExecution
	D	SX001G:systemPartialMissionCapableDuringTaskExecution
	E	SX001G:productNotMissionCapableDuringTaskExecution
	G	SX001G:turnaroundTask
taskPersonnelResourceRole	A	SX001G:assistentTaskPersonnelResource
	P	SX001G:performerTaskPersonnelResource
	Q	SX001G:qualityAssuranceTaskPersonnelResource
	S	SX001G:supervisorTaskPersonnelResource
taskPersonnelSafetyCriticalit	1	SX001G:catastrophicPersonnelSafetyCriticality

Attribute name	Valid value	Valid value name
y	2	SX001G:criticalPersonnelSafetyCriticality
	3	SX001G:marginalPersonnelSafetyCriticality
	4	SX001G:negligiblePersonnelSafetyCriticality
taskProductIntegrityCriticality	N	SX001G:nonCriticalProductIntegrityTask
	Y	SX001G:criticalProductIntegrityTask
taskRequirementAuthoritySourceType	CMR	SX001G:certificationMaintenanceRequirement
	EZAP	SX001G:enhancedZonalAnalysisProgram
	LEMR	SX001G:lawEnforcedMaintenanceRequirement
	PMEC	SX001G:preventiveMaintenanceEffectCategory
taskRequirementIdentifier	ID	SX001G:taskRequirementIdentifier
taskRequirementInformationCode		Refer to taskInformationCodeValues valid value library, Para 5.
taskRequirementRevisionIdentifier	ID	SX001G:taskRequirementRevisionIdentifier
taskRequirementRevisionStatus		Refer to revisionStatusCode valid value library, Para 5.
taskResourceIdentifier	ID	SX001G:taskResourceIdentifier
taskResourceRelationshipType	AND	SX001G:mutualInclusionTaskResourceRelationship
	COOP	SX001G:cooperationTaskResourceRelationship
	SEQ	SX001G:sequenceTaskResourceRelationship
	USE	SX001G:usesTaskResourceRelationship
	XOR	SX001G:mutualExclusionTaskResourceRelationship
taskRevisionIdentifier	ID	SX001G:taskRevisionIdentifier
taskRevisionStatus		Refer to revisionStatusCode valid value library, Para 5.
technologyBehaviourKnowledgeRating	1	SX001G:wellKnownTechnologyBehavior
	2	SX001G:knownTechnologyBehavior
	3	SX001G:unknownTechnologyBehavior
technologySensitivityRating	1	SX001G:extremelyLowLikelihoodOfDamage
	2	SX001G:lowLikelihoodOfDamage
	3	SX001G:mediumLikelihoodOfDamage
	4	SX001G:highLikelihoodOfDamage
	5	SX001G:extremelyHighLikelihoodOfDamage
thresholdValueQualifier	A	SX001G:executeAfterThresholdValueDefinition

Attribute name	Valid value	Valid value name
	B	SX001G:executeBeforeThresholdValueDefinition
warningCautionNoteIdentifier	ID	SX001G:warningCautionNoteIdentifier
warningCautionNoteType	C	SX001G:cautionAdvise
	N	SX001G:noteAdvise
	W	SX001G:warningAdvise
zoneElementType	W	SX001G:productWorkArea
	Z	SX001G:productZone

5 Valid value libraries

The list of valid value libraries used by S3000L is provided in Table 5. These libraries include codes from international specifications. Details on the valid values included in the respective valid value library are given in the valid value XML files that accompany this specification.

Note

Specified valid values are recommended values and projects can therefore tailor the valid values as needed (refer to SX005G).

Table 5 List of valid value libraries

Library name	Valid value	Valid value name
comparisonOperatorCodeValues	EQ	SX001G:equalToComparisonOperator
	NE	SX001G:notEqualToComparisonOperator
	LT	SX001G:lessThanComparisonOperator
	LE	SX001G:lessThanOrEqualToComparisonOperator
	GT	SX001G:greaterThanComparisonOperator
	GE	SX001G:greaterThanOrEqualToComparisonOperator
	IN	SX001G:withinRangeComparisonOperator
	OUT	SX001G:outsideRangeComparisonOperator
countryCodeValues	-	**Note** The country codes valid value XML schema for S3000L issue 2.0 lists 193 county codes from ISO 3166-1.
hardwarePartHazardousClassCodeValues	-	**Note** The hazardous class code valid value XML schema does not include any predefined valid values.
languageCodeValues	-	**Note** The language codes valid value XML schema for S3000L issue 2.0 lists 136 language codes from ISO 639:1988.
revisionStatusCodeValues	A	SX001G:approvedStatus

Library name	Valid value	Valid value name
	D	SX001G:draftStatus
	IW	SX001G:inWorkStatus
	C	SX001G:cancelledStatus
	R	SX001G:reviewedStatus
taskInformationCodeValues	-	**Note** The task information codes valid value XML schema for S3000L issue 2.0 lists 41 information codes from S1000D,