Date: February 2026

CASCaRA

Collaborative Artifact, Specification, Context and Resource Access in response to the CASCaDE RFP (mantis/2024-12-03)

Version 1.2

OMG Document Number: mantis/2026-02-02

Standard Document URL: https://www.omg.org/spec/cascara/1.2/

USE OF SPECIFICATION - TERMS, CONDITIONS & NOTICES

The material in this document details an Object Management Group specification in accordance with the terms, conditions and notices set forth below. This document does not represent a commitment to implement any portion of this specification in any company's products. The information contained in this document is subject to change without notice.

LICENSES

The companies listed above have granted to the Object Management Group, Inc. (OMG) a nonexclusive, royalty-free, paid up, worldwide license to copy and distribute this document and to modify this document and distribute copies of the modified version. Each of the copyright holders listed above has agreed that no person shall be deemed to have infringed the copyright in the included material of any such copyright holder by reason of having used the specification set forth herein or having conformed any computer software to the specification.

Subject to all of the terms and conditions below, the owners of the copyright in this specification hereby grant you a fully-paid up, non-exclusive, nontransferable, perpetual, worldwide license (without the right to sublicense), to use this specification to create and distribute software and special purpose specifications that are based upon this specification, and to use, copy, and distribute this specification as provided under the Copyright Act; provided that: (1) both the copyright notice identified above and this permission notice appear on any copies of this specification; (2) the use of the specifications is for informational purposes and will not be copied or posted on any network computer or broadcast in any media and will not be otherwise resold or transferred for commercial purposes; and (3) no modifications are made to this specification. This limited permission automatically terminates without notice if you breach any of these terms or conditions. Upon termination, you will destroy immediately any copies of the specifications in your possession or control.

PATENTS

The attention of adopters is directed to the possibility that compliance with or adoption of OMG specifications may require use of an invention covered by patent rights. OMG shall not be responsible for identifying patents for which a license may be required by any OMG specification, or for conducting legal inquiries into the legal validity or scope of those patents that are brought to its attention. OMG specifications are prospective and advisory only. Prospective users are responsible for protecting themselves against liability for infringement of patents.

GENERAL USE RESTRICTIONS

Any unauthorized use of this specification may violate copyright laws, trademark laws, and communications regulations and statutes. This document contains information which is protected by copyright. All Rights Reserved. No part of this work covered by copyright herein may be reproduced or used in any form or by any means--graphic, electronic, or mechanical, including photocopying, recording, taping, or information storage and retrieval systems--without permission of the copyright owner.

DISCLAIMER OF WARRANTY

WHILE THIS PUBLICATION IS BELIEVED TO BE ACCURATE, IT IS PROVIDED "AS IS" AND MAY CONTAIN ERRORS OR MISPRINTS. THE OBJECT MANAGEMENT GROUP AND THE COMPANIES LISTED ABOVE MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS PUBLICATION, INCLUDING BUT NOT LIMITED TO ANY WARRANTY OF TITLE OR OWNERSHIP, IMPLIED WARRANTY OF MERCHANTABILITY OR WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE OR USE. IN NO EVENT SHALL THE OBJECT MANAGEMENT GROUP OR ANY OF THE COMPANIES LISTED ABOVE BE LIABLE FOR ERRORS CONTAINED HEREIN OR FOR

DIRECT, INDIRECT, INCIDENTAL, SPECIAL, CONSEQUENTIAL, RELIANCE OR COVER DAMAGES, INCLUDING LOSS OF PROFITS, REVENUE, DATA OR USE, INCURRED BY ANY USER OR ANY THIRD PARTY IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS MATERIAL, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

The entire risk as to the quality and performance of software developed using this specification is borne by you. This disclaimer of warranty constitutes an essential part of the license granted to you to use this specification.

RESTRICTED RIGHTS LEGEND

Use, duplication or disclosure by the U.S. Government is subject to the restrictions set forth in subparagraph (c) (1) (ii) of The Rights in Technical Data and Computer Software Clause at DFARS 252.227-7013 or in subparagraph (c)(1) and (2) of the Commercial Computer Software - Restricted Rights clauses at 48 C.F.R. 52.227-19 or as specified in 48 C.F.R. 227-7202-2 of the DoD F.A.R. Supplement and its successors, or as specified in 48 C.F.R. 12.212 of the Federal Acquisition Regulations and its successors, as applicable. The specification copyright owners are as indicated above and may be contacted through the Object Management Group, 9C Medway Road PMB 274, Milford, MA, 01757 U.S.A.

TRADEMARKS

CORBA®, CORBA logos®, FIBO®, Financial Industry Business Ontology®, FINANCIAL INSTRUMENT GLOBAL IDENTIFIER®, IIOP®, IMM®, Model Driven Architecture®, MDA®, Object Management Group®, OMG®, OMG Logo®, SoaML®, SOAML®, SysML®, UAF®, Unified Modeling Language®, UML®, UML Cube Logo®, VSIPL®, and XMI® are registered trademarks of the Object Management Group, Inc.

For a complete list of trademarks, see: https://www.omg.org/legal/tm_list.htm. All other products or company names mentioned are used for identification purposes only, and may be trademarks of their respective owners.

COMPLIANCE

The copyright holders listed above acknowledge that the Object Management Group (acting itself or through its designees) is and shall at all times be the sole entity that may authorize developers, suppliers and sellers of computer software to use certification marks, trademarks or other special designations to indicate compliance with these materials.

Software developed under the terms of this license may claim compliance or conformance with this specification if and only if the software compliance is of a nature fully matching the applicable compliance points as stated in the specification. Software developed only partially matching the applicable compliance points may claim only that the software was based on this specification, but may not claim compliance or conformance with this specification. In the event that testing suites are implemented or approved by Object Management Group, Inc., software developed using this specification may claim compliance or conformance with the specification only if the software satisfactorily completes the testing suites.

OMG's Issue Reporting Procedure

All OMG specifications are subject to continuous review and improvement. As part of this process we encourage readers to report any ambiguities, inconsistencies, or inaccuracies they may find by completing the Issue Reporting Form listed on the main web page http://www.omg.org, under Specifications, Report a Bug/Issue.

Table of Contents

Preface

OMG

Founded in 1989, the Object Management Group, Inc. (OMG) is an open membership, not-for-profit computer industry standards consortium that produces and maintains computer industry specifications for interoperable, portable, and reusable enterprise applications in distributed, heterogeneous environments. Membership includes Information Technology vendors, end users, government agencies, and academia.

OMG member companies write, adopt, and maintain its specifications following a mature, open process. OMG's specifications implement the Model Driven Architecture® (MDA®), maximizing ROI through a full-lifecycle approach to enterprise integration that covers multiple operating systems, programming languages, middleware and networking infrastructures, and software development environments. OMG's specifications include: UML® (Unified Modeling Language™); CORBA® (Common Object Request Broker Architecture); CWM™ (Common Warehouse Metamodel™); and industry-specific standards for dozens of vertical markets.

More information on the OMG is available at http://www.omg.org/.

OMG Specifications

As noted, OMG specifications address middleware, modeling and vertical domain frameworks. All OMG Specifications are available from the OMG website at:

https://www.omg.org/spec

All of OMG's formal specifications may be downloaded without charge from our website. (Products implementing OMG specifications are available from individual suppliers.) Copies of specifications, available in PostScript and PDF format, may be obtained from the Specifications Catalog cited above or by contacting the Object Management Group, Inc. at:

OMG Headquarters
9C Medway Road, PMB 274
Milford, MA 01757
USA
Tel: +1-781-444-0404
Fax: +1-781-444-0320
Email: *pubs@omg.org*

Certain OMG specifications are also available as ISO standards. Please consult https://www.iso.org

1. Submission Introduction

1.1. Submission Overview

CASCaRA (Collaborative Artifact, Specification, Context and Resource Access) is the response to the OMG CASCaDE RFP (mantis/2024-12-3).

This document describes a metamodel and layered ontology approach to semantic integration and interoperability of engineering information and data involved in engineering collaboration. More concretely, the metamodel governs the CASCaRA Product Information Graph (PIG) that serves as the basis for representation of the integrated information.

Additionally, CASCaRA provides an API to operate on the integrated product information and defines a mechanism for transformation of information and data from source formats to the PIG.

1.2. Submitters to the CASCaDE RFP

The OMG members listed below are jointly submitting the CASCaRA specification in response to the CASCaDE RFP:

- Adaptive Analytics, Inc.
- BOEING
- GfSE eV
- INCOSE
- prostep ivip eV
- RTX

Many other organizations worldwide supported the development of CASCaRA in the CASCaDE Submitters Team (**CST**).

1.3. Issues to be discussed

In this section the open issues of the initial submission are discussed. For the final submission this section should be empty.

1.4. RFP requirements table

1.4.1. Mandatory Language Requirements Table

Table 0.1: Mandatory RFP requirements

Req. ID	Name	Content
MM-001	Universal Metamodel	The metamodel shall universally host a range of domain ontologies used in the field of Product Lifecycle Collaboration. Separating the concerns of syntax and semantics, the evolution of the ontology (defining entities, relations and their properties) shall not entail a change in the metamodel.
MM-002	Data is Represented as a Graph	Proposals shall represent product information as a Knowledge Graph.
MM-003	Formal Metamodel	Proposals shall formally define the metamodel (aka schema). Established technology shall be used for

Req. ID	Name	Content
		• checking data to ingest, often transformed from one of the supported data-formats, or to deliver. • generating software code in many programming languages.
MM-003-01	Model Entities	Proposals shall provide a capability to represent individual entities obtained from a source model with their properties and provide a distinct representation for each relevant entity from given source data.
MM-003-02	Model Relations	Proposals shall provide the capability to represent individual relations between entities obtained from a source model with their properties. Similarly as model entities, model relations shall have a distinct representation for each relevant relation from given source data.
MM-003-03	Model Elements have Types	The metamodel shall not only cater the instantiation of model entities and relations, but also of explicit types (classes) for both. In addition the metamodel shall provide a means to define which types of relations can be applied to which types of entities on either side.
MM-003-04	Validation of Models	Proposals shall provide capabilities to validate a CASCaDE Data Package (CDP) based on its metamodel (schema) and constraints.
OP-001	Conformance with Web-Standards	Data Exchange and Invocation of Operations shall follow the standards governing the world-wide-web (internet) from W3C and others.
OP-002	Standard Application Programming Interface (API)	Data Exchange and Invocation of Operations shall follow the standards governing the world-wide-web (internet) from W3C and others.
OP-002-01	Mutable Model-elements	Any entity and relation with their properties as well as their types/classes shall be accessible and mutable with elementary operations Create, Read, Update and Delete (CRUD).
OP-002-02	Access Control	Access to Product Information shall be protected. Parts of a model which are not intended to be shared, shall be withheld from the CDP. Access permissions to Create, Read, Update and Delete (CRUD) shall be granted • per Organization and User on one hand as well as • per Package, Hierarchy and Type of Entity, Relation and Property on the other hand. Thus, access shall be controlled in two dimensions per class at the granularity of a property and per position in the BOM or document outline.
OP-003	Configuration Management	Any model-element (node and edge of the knowledge graph) shall be under configuration control. This includes management of revisions and tags for selected baselines. Version control for entities and relations shall provide at least the following operations: • create revision (automatically on modification), • create a baseline with label/tag (deliberately on a certain milestone), • retrieve any (historical and current) revision of a model-element or baseline as a whole.
OP-004	Distributed Configuration Management	Based on configuration control, the solution proposal shall have at least the capability to • create and merge a branch as well as • fork and synchronize a repository/package.

Req. ID	Name	Content
OP-005	Globalize Term	Terms in specific (engineering) disciplines and natural languages shall be transformed to a unique ('global') term as defined by the layered ontology.
OP-006	Localize Term	Unique ('global') terms shall be transformed to specific (engineering) disciplines or natural languages as defined by the layered ontology.
OP-007	Normalize Term	Unique ('global'), yet synonymous terms shall be mapped to a preferred ('normalized') term as defined by the layered ontology. In case of relations, the direction shall be normalized. An antonym shall be used in certain cases to obtain the preferred direction (example: replace 'satisfied by' relation by 'satisfies').
OP-008	Integrate Partial Models	Partial models shall be integrated in the Knowledge Graph.
OP-009	Extract Product Information	When viewing or exporting product information, the scope shall be selectable for the purpose ('need to know') and within the limits of access permissions of the target organization or user. This is also called 'publishing' product information.
OP-009-01	Extract by Type	Similarly to the definition of access protection, proposals shall provide the ability to limit extraction to certain entity, relation or property types.
OP-009-02	Extract by Hierarchy	Similarly to the definition of access protection, proposals shall provide the ability to limit extraction to certain subtrees of the hierarchy (outline or BoM).
OP-009-03	Extract by Instance	Proposals shall provide the ability to extract product information by selecting or deselecting individual entities, relations and properties.
OP-010	Human-readable Content	Product information shall be provided human readable where possible. Preferred data formats are those displayable by a web-browser, such as HTML, SVG, PNG and JPG; acceptable is PDF.
OP-011	Overview	A human-readable content overview (outline) shall be provided as an entry point. The human-readable overview shall reflect the structure of the package content: • Header data, including package metadata, with relation to model entities • Source models and metadata • Comments related to model entities • Model entities, relations with their properties • Any other original sources
OP-012	Share and Review in a Common Context	Proposals shall provide the ability to share and review all product information in a common context.
OP-013	Search and Navigate 'Globally'	Users and interfaced systems shall be able to navigate, search and query the whole product information (Knowledge Graph), as far as access control permits. This imposes certain requirements on the transformations of partial models to form a consistent and homogenous (typed) knowledge graph.
OP-014	Collaborate Online and Offline	Proposals shall support both online and offline collaboration, where • 'online' or 'synchronous' means a multi-user editing on a single database, the 'Collaboration Hub'; • 'offline' or 'asynchronous' means information exchange by file. File based exchange shall support and editing on different branches in parallel and subsequent merging of changes.

Req. ID	Name	Content
DO-001	Ontology based on semantic data-model	The solution shall provide a layered ontology based on a widely used information model to be defined.
DO-001-01	Layered Ontology Coverage	Submissions shall provide a layered ontology covering: • Process activities along the product lifecycle with a specification of entities (artifacts) provided or required by the respective activity. The technical processes defined in ISO 15288 shall be used as a reference. • Type/class per entity (artifact) such as requirement, function and component, as well as its property classes. • Type/class per relation including a definition which entity types/classes can be related, as well as its property classes. • Type/class per property including a definition of data type and value range.
DO-001-02	Synonyms and Antonyms	The layered ontology shall contain synonym and antonym relationships between entries to support normalizing of terms.
DO-001-03	National Languages	The layered ontology shall contain national language and/or domain-specific terms per entry to support localizing and globalizing of terms.
DO-001-04	Ontology Lifecycle Management	Submissions shall support the evolution of the layered ontology over time, yet allow for reliable and robust use of past and current states. Therefore, classes, relations and properties in the ontology shall have a revision and a lifecycle status such as *experimental, submitted, released, equivalent, deprecated*. If a standard exists for such term status values, it shall be adopted.
DO-002	Information Packaging	Proposals shall support a segmentation of the overall product information into nested packages.
DO-003	Unstructured Artifacts	Some artifacts consist of partly or wholly unstructured content. This usually applies to simulation models and simulation results, as well as media documents such as images or videos. For such data, a capability shall be provided to handle unstructured data and ingest these artefacts as-is.
DO-004	Handle Model Meta-Data	Besides handing the entities and relations extracted from the source models, it is also required to understand its context, e. g. origin, purpose, maturity, etc. Such metadata is typically available from the respective data management systems and shall be supplied as well, if available.
DO-005	Visibility	Proposals shall distinguish public and private elements. Private elements can be referenced from within a package, and public elements can be referenced externally or internally.
DO-006	Include or Reference Source Files	Proposals shall provide the ability to include notation- or tool-specific source files in addition to the transformed data. Source files shall either be included as a copy or referenced by a URL.
DO-007	Commenting	For auditing or negotiation purposes, users and interfaced systems shall be able to attach comments to any entity or relation without changing the entities or relations themselves. A comment shall be always owned by a user and relate to one or several model entities, as well as any previous comment.
DO-007-01	SRC Attributes	Proposals shall provide annotations to capture information originating from the the Stakeholder Request Clarification (SRC) process, such as, but not limited to: • customer status with defined values • customer comment • supplier status with defined values

Req. ID	Name	Content
		• supplier comment.
DO-008	Semantic Integration	Capability to recognize and relate or combine identical entities from different source models, even from different notations or formats. Also recognize and integrate relations between entities found in different source models.
DO-008-01	Unify Entities	The specification shall provide the capability to relate or unify 'same' entities from different source models.
DO-008-02	Extracted and Created Relations	Submissions shall provide the ability to represent relations extracted from source models or manually added relations. Any relation shall be consistent with its type and thus be constrained to the defined resource types for the originating entity (subject) and the targeted entity (object).
TR-001	Transform Source Formats	Proposals shall support (partial) transformations between multiple source representations and a common representation as Knowledge Graph. A unidirectional transformation or filtering from a source format to the Knowledge Graph is mandatory, whereas a bidirectional transformation shall be proposed only if a full round-trip is required. The transformation shall be limited to those entities and relations within a source format which are relevant for collaboration or overarching assessment. The primary goal is to create a common representation for entities and relations from (partial) source models, allowing for an end-to-end 'digital thread'. Therefore proposals shall provide full or partial transformations with or without filtering for the following technologies:
TR-001-01	Transform ReqIF	
TR-001-02	Transform SysML v1	
TR-001-03	Transform SysML v2	
TR-001-04	Transform STEP AP242	
TR-001-05	Transform JT	
TR-001-06	Transform FMI 3.0 and SSP	
TR-001-07	Transform Modelica	
TR-001-08	Transform QIF	
TR-002	Preserve Views contained in Source Models	Proposed transformations shall preserve view and viewpoint information contained in source models wherever possible. This shall include but not limited to diagrams of UML, SysML, BPMN, and 2D, and 3D models. Proposals shall provide a capability to display such views displayable by web-browsers.

Table 0.2: Non-Mandatory RFP requirements

Req. ID	Name	Content
TR-001-09	Transform Arcadia	Solutions may provide a transformation of system models from the Architecture Analysis & Design Integrated Approach (ARCADIA) notation.
TR-001-10	Transform BPMN	Solutions may provide a transformation of system models from the BPMN v2 notation using the standardized BPMN-XML data format.
TR-001-11	Transform ArchiMate	Solutions may provide a transformation of system models from the ArchiMate v3 notation using the standardized Open-Exchange data format.

1.4.2. Mandatory Language Requirements - Satisfied-by Table

Req. ID	Name	Satisfied by	Comment
MM-001	Universal Metamodel	Designed for purpose	Not validated, yet.
MM-002	Data is Represented as a Graph	Designed for purpose	
MM-003	Formal Metamodel	Designed as UML model.	No implementation for format checking and code generation, yet.
MM-003-01	Model Entities	Designed for purpose	Not validated, yet.
MM-003-02	Model Relations	Designed for purpose, relationships are reified to allow ‘statements on statements’ and ‘properties of statements’.	Not validated, yet.
MM-003-03	Model Elements have Types	Designed for purpose; the classes or types are derived from the ontology, whereas the the instances hold the payload and must comply with the types.	Not validated, yet.
MM-003-04	Validation of Models	Designed for purpose	Not implemented, yet.
OP-001	Conformance with Web-Standards		
OP-002	Standard Application Programming Interface (API)		
OP-002-01	Mutable Model-elements		
OP-002-02	Access Control		
OP-003	Configuration Management		
OP-004	Distributed Configuration Management		
OP-005	Globalize Term		
OP-006	Localize Term		
OP-007	Normalize Term		

Req. ID	Name	Satisfied by	Comment
OP-008	Integrate Partial Models		
OP-009	Extract Product Information		
OP-009-01	Extract by Type		
OP-009-02	Extract by Hierarchy		
OP-009-03	Extract by Instance		
OP-010	Human-readable Content		
OP-011	Overview		
OP-012	Share and Review in a Common Context		
OP-013	Search and Navigate 'Globally'		
OP-014	Collaborate Online and Offline		
DO-001	Ontology based on semantic data-model		
DO-001-01	Layered Ontology Coverage		
DO-001-02	Synonyms and Antonyms		
DO-001-03	National Languages		
DO-001-04	Ontology Lifecycle Management		
DO-002	Information Packaging		
DO-003	Unstructured Artifacts		
DO-004	Handle Model Meta-Data		
DO-005	Visibility		
DO-006	Include or Reference Source Files		
DO-007	Commenting		
DO-007-01	SRC Attributes		
DO-008	Semantic Integration		
DO-008-01	Unify Entities		
DO-008-02	Extracted and Created Relations		
TR-001	Transform Source Formats		

Req. ID	Name	Satisfied by	Comment
TR-001-01	Transform ReqIF		
TR-001-02	Transform SysML v1		
TR-001-03	Transform SysML v2		
TR-001-04	Transform STEP AP242		
TR-001-05	Transform JT		
TR-001-06	Transform FMI 3.0 and SSP		
TR-001-07	Transform Modelica		
TR-001-08	Transform QIF		
TR-002	Preserve Views contained in Source Models		

2. Scope

CASCaRA (Collaborative Artifact, Specification, Context and Resource Access) was elaborated in response to the CASCaDE RFP (mantis/2024-12-3) which puts the digital engineering collaboration in the focus of consideration. One key problem to be addressed for a seamless collaboration between engineers is the semantic integration of information connected with the product.

CASCaRA defines a metamodel for a graph-based representation of information that abstracts from the semantic representation using an ontology-driven approach to formally describe the meaning of artifacts and their relationships. A design principle of the specification is to separate the syntax (metamodel) from the semantics (ontology) to ensure a stable and resilient implementation of the specification while the semantics may evolve over time and new ontologies may be included for the extension of the standard to other domains.

Additionally, CASCaRA defines a API to allow external operations to connect to the graph in support of queries and analyses. CASCaRA will also provide a limited set of transformations (mappings) from source formats like ReqIF, SysML, 3D CAD (JT, STEP,…), and Office Formats (Excel, …).

The following scenarios are considered to allow individual preferences and infrastructure:

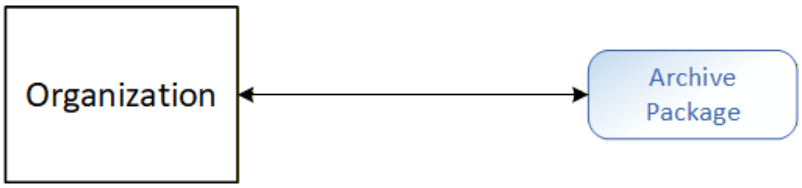


Figure 1 Long-term Archiving

Organizations primarily in regulated industries need to archive engineering data at least for the lifetime of a product, which may last many decades. Data requires a standardized format and ontology without the need for special software.

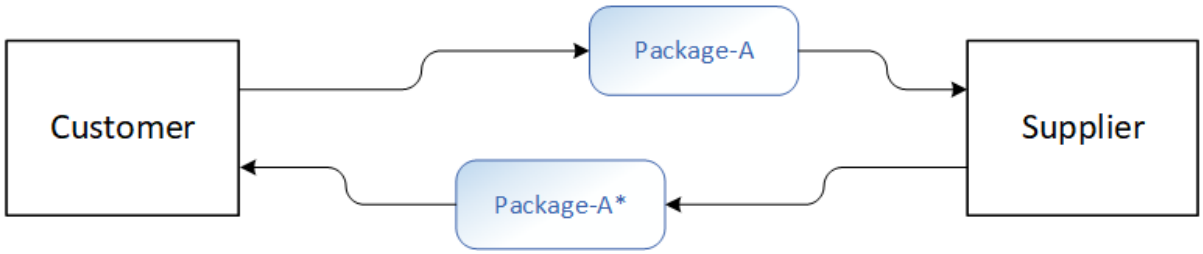


Figure 2 Collaboration Mode 1: Traditional File Transfer

Some organizations resist permitting access to their IT infrastructure to others and prefer to exchange selected information by file. In Collaboration Mode 1 partners operate on separate databases and synchronize their data periodically.

Figure 3 Collaboration Mode 2: One partner provides Hub

In collaboration mode 2 one of the partners offers a common database for synchronous transaction processing. The remote partner receives permission to access the data of interest usually differentiated for create, read, update or delete operations.

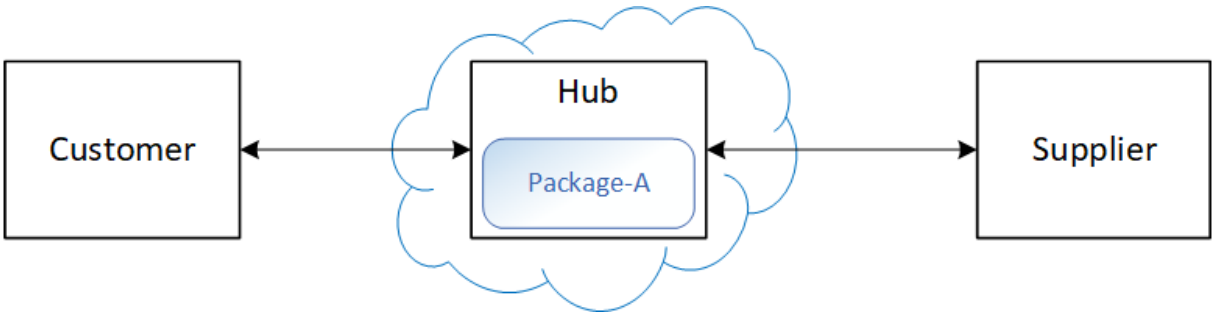


Figure 4 Collaboration Mode 3: Hub as a Public Service

Collaboration Mode 3 is very similar to the previous scenario, except that a common database operated by a third party („SaaS“) is used.

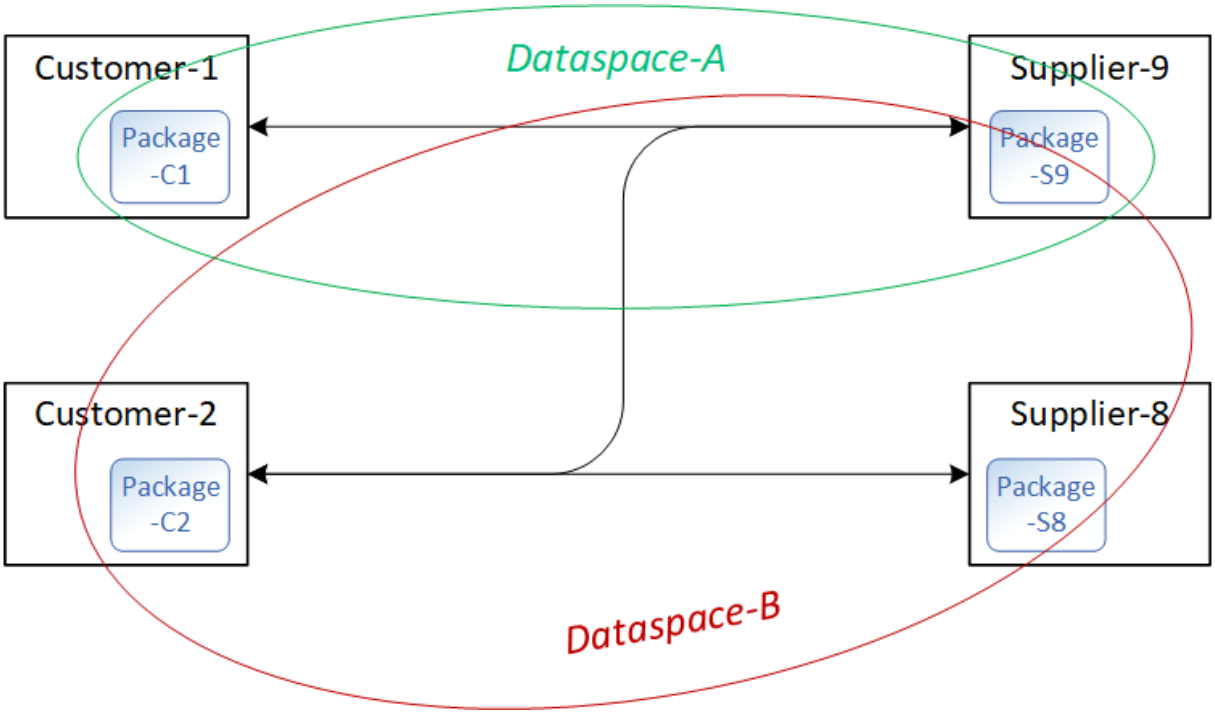


Figure 5 Collaboration Mode 4: Federated Dataspace

In the most advanced scenario, each partner retains full control of her or his data and permits others to access to data of interest. Dataspaces with distributed databases are created for transaction processing. A partner‘s package may be part of multiple dataspaces.

3. Conformance

This section is TBD

4. Normative References

The following normative documents contain provisions which, through reference in this text, constitute provisions of this specification. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply.

[KerML] Kernel Modeling Language (KerML), Version 1.0

[MOF] Meta Object Facility, Version 2.5.1
https://www.omg.org/spec/MOF/2.5.1

[OCL] Object Constraint Language, Version 2.4
https://www.omg.org/spec/OCL/2.4

[SMOF] MOF Support for Semantic Structures, Version 1.0
https://www.omg.org/spec/SMOF/1.0

[SysMLv1] OMG Systems Modeling Language (SysML), Version 1.7
(currently in preparation)

[UML] Unified Modeling Language (UML), Version 2.5.1
https://www.omg.org/spec/UML/2.5.1

Systems Modeling Language (SysML) v2
https://www.omgsysml.org/SysML-2.htm

Systems Modeling API and Services
https://www.omg.org/spec/SystemsModelingAPI

Requirement Interchange Format (ReqIF)
https://www.omg.org/spec/ReqIF/

Risk Analysis and Assessment Modeling Language (RAAML)
https://www.omg.org/spec/RAAML/1.0/About-RAAML: The RAAML Version 1.0 specification defines extensions to SysML needed to support safety and reliability analysis.

Business Process Model and Notation (BPMN)
https://www.omg.org/spec/BPMN can be used to capture the processes used in preparing, maintaining, and using datasets – for the purpose of traceability and repeatability.

Pedigree and Provenance Model and Notation (PPMN)
https://www.omg.org/spec/PPMN can be used to capture the derivation and chain of custody of a dataset

Ontology Definition Metamodel (ODM)
https://www.omg.org/spec/ODM can be used to represent data structures in ontology and RDF formats

Commons Ontology Library
https://www.omg.org/spec/COMMONS provides a set of reusable ontology patterns for common structures such as identifiers

Distributed Ontology, Model, and Specification Language (DOL)
https://www.omg.org/dol/ gives interoperability a formal grounding and makes heterogeneous OMS and services based on them amenable to checking of coherence (e.g., consistency, conservativity, intended consequences, and compliance).

Multiple Vocabulary Facility (MVF)
https://www.omg.org/spec/MVF: Providing structures for representing glossaries/vocabularies as used by different communities for common data elements.

Enterprise Knowledge Graph
https://www.omg.org/ekg/: Shaping the future of enterprise knowledge management through collaboration and innovation. At the forefront of this effort is our commitment to codifying industry consensus and best practices into OMG specifications, reference models, and discussion papers.

APIs for Knowledge Platforms
https://www.omg.org/spec/API4KP/1.0/About-API4KP: This OMG specification defines the Application Programming Interfaces for Knowledge Based Systems and Platforms (API4KP). The purpose of these APIs is to facilitate the development and integration of knowledge graphs and knowledge-based systems in a broader enterprise framework.

Common Warehouse Metamodel (CWM)
https://www.omg.org/spec/CWM: The CWM can be used to represent data structures in different enterprise formats, including tabular/relational and XML; and its extraction and transformation (ETL) from original sources.

System Package Data Exchange (SPDX)
https://www.omg.org/spec/SPDX/3.0/Beta1/PDF: A data exchange format so that information about system packages and related content (such as software, data sets, AI models, security, licencing, and build information) may be collected and shared in a common format with the goal of saving time and improving data accuracy.

Essence
https://www.omg.org/spec/Essence/2.0/Beta1/About-EssenceThe Kernel provides the common ground for defining software development practices.

Multiple Vocabulary Facility (MVF)
https://www.omg.org/spec/MVF/1.0/About-MVF: OMG modeling communities include people that may communicate in different natural or specialized languages. Often the patterns represented in models are reusable, either directly or with minor differences across organizations with a business, discipline, or domain area. For effective interaction and user understanding, models shall be expressed in natural languages and sometimes nomenclatures or in terms of other business vocabulary appropriate for the intended users.

Information Exchange Packaging Policy Vocabulary (IEPPV)
https://www.omg.org/spec/IEPPV/1.0/About-IEPPV: The vocabulary is intended to improve the accuracy, fidelity, clarity, and consistency in the specification and design of data assembly and processing rules corresponding to information sharing and safeguarding (ISS) policies.

Data Product Ontology (DPROD):
https://ekgf.github.io/dprod/ : The concept of Data Products has emerged as organizations increasingly recognize the value of data as an asset to be managed and distributed like any other product. As more companies adopt decentralized data architectures, such as Data Mesh, the need for standardized methods to describe and manage data products consistently across platforms has become critical.

5. Terms and Definitions

For the purposes of this specification, the following terms and definitions apply.

Artifact - According to ISO-15288 an ***Artifact*** is a work product that is produced and used during a project to capture and convey information. An artifact may be fine-grained such as a model-element or more coarse-grained such as business

data object or a data set. Please note that in this document the spelling ‘artifact’ is chosen in accordance with American English and OMG preference, whereas ISO-15288 uses the spelling ‘artefact’.

A ***Knowledge Graph*** represents knowledge in a structured form, storing entities and their relationships in a graph and supporting reasoning over the data. RDF provides the triple-based structure (subject-predicate-object) commonly used in knowledge graphs..

Sources: Paulheim, H. (2017): "Knowledge Graph Refinement: A Survey of Approaches and Evaluation Methods"., "Knowledge Graphs: Fundamentals, Techniques, and Applications" (by Dieter Fensel et al.)” and various vendors.

Property Graph is a graph data model designed to represent and manage complex, interconnected data. It is characterized by nodes (entities or objects) and edges (relationships between nodes), where both nodes and edges can have associated properties in the form of key-value pairs..

Source: "Graph Databases" by Ian Robinson, Jim Webber, and Emil Eifrem.

Product Information or ***System Model*** describes the product at MOF level M0.

Instance – Used synonymously to Object and Entity.

Information Model is the metamodel of a system model, thus at MOF level M01. It is defined by schema and constraints.

A ***Schema*** is a means of representing an Information Model.

A ***source model*** or ***partial model*** consists of entities, attributes and relations, following a specific schema or metamodel.

A ***Model-element*** is the most general, atomic element of a system model.

Entity - An ***Entity*** or ***Resource*** or ***Object*** is an element of the system model (thus a model-element) or node in the knowledge graph.

A ***Relation*** is an element of the system model (thus a model-element as well) connecting two resources/entities or edge in the knowledge graph. Relation is the predicate in a statement.

A ***Statement*** or ***Triple*** corresponds to a pair of resources with a relation used as a logic assertion.

The ***CASCaDE Data Package*** (CDP) stands for concrete product information according to the CASCaDE metamodel and layered ontology. It may apply to serialized data in a file or to data persisted in a database. Furthermore it can apply to the full data-set or an extracted subset. Important is that its syntax and semantics comply with the CASCaDE standard.

A ***Term*** is the name of cognitive concept, thus a material or immaterial *thing*.

A ***Vocabulary*** is a list of controlled terms with a definition representing model-element types.

A ***Taxonomy*** is a model whose structure (hierarchy) specifies generalization/ specialization relationships.

Source: ISO/IEC 11179-3: Information technology — Metadata registries (MDR) — Part 3: Registry metamodel and basic attributes, Third edition, 2013-02-15

An ***Ontology*** is a model that specifies a rich description of the

- terminology, concepts, nomenclature;
- relationships among and between concepts and individuals; and
- sentences distinguishing concepts, refining definitions and relationships (constraints, restrictions, regular expressions) relevant to a particular domain or area of interest.

Source: Elisa F. Kendall, Deborah L. McGuinness. Ontology Engineering. Synthesis Lectures on the Semantic Web: Theory and Technology, Ying Ding and Paul Groth, Editors. Morgan & Claypool, 2019.

6. Symbols

Non specific in this document.

7. Additional Information

7.1. Changes to Adopted OMG Specifications [optional]

None.

7.2. Acknowledgments

The following companies submitted this specification:

- Adaptive Analytics, Inc.
- Boeing
- GfSE eV
- INCOSE
- Object Management Group
- ProSTEP iViP Association
- RTX

The organizations listed below contributed to work in the submission of this specification:

- :em Engineering Methods AG
- Aerospace Corp.
- Airbus
- ATLAS Elektronik
- Collins Aerospace
- enso managers
- Federated Knowledge, LLC
- HSU
- Inomic Solutions
- Manfred Harms Consulting
- ModelAlchemy Consulting
- MTSI
- oose
- PDES Inc.
- PSG Inc.
- PROSTEP AG
- RPTU
- Schaeffler
- Thematix Partners
- TU Ilmenau
- TUHH
- UC3 Madrid
- VDA

8. Specification

8.1. Overview

CASCaRA responds to the CASCaDE RFP requirements to define and specify the following:

- A **metamodel** with schema and constraints to represent product information conforming to domain specific ontologies
- A concept for a **layered ontology** approach to semantically integrate the product information conforming to the metamodel
- A set of **operations** in support of synchronous and asynchronous information sharing
- A set of **transformations** from source formats to the metamodel.

In general, CASCaRA chooses a graph-based approach for the representation of information where different formats like RDF, JSON-LD and property graphs are under consideration. A decision was made to define a metamodel that was able to support all graph representations to include a wide range of technologies for implementations.

8.2. CASCaRA Metamodel

8.2.1. CASCaRA Metamodel Diagram

The following major design goals are pursued:

- Separate syntax and semantics: The metamodel governs the graph pattern independently of the ontology. Any software depends on the metamodel and doesn't need updating, when the ontology evolves over time.
- Simplify transformation: A graph pattern is designed which may be represented in RDF, JSON-LD, GQL and OOP alike. Bidirectional transformations between any of those formats shall be possible without data-loss. Admittedly this 'sweet-spot' has reduced expressivity when compared with any of the listed data formats, but the choice of implementation technologies is maximized. Verification and validation will show that the resulting expressivity of the graph is sufficient for the purpose.

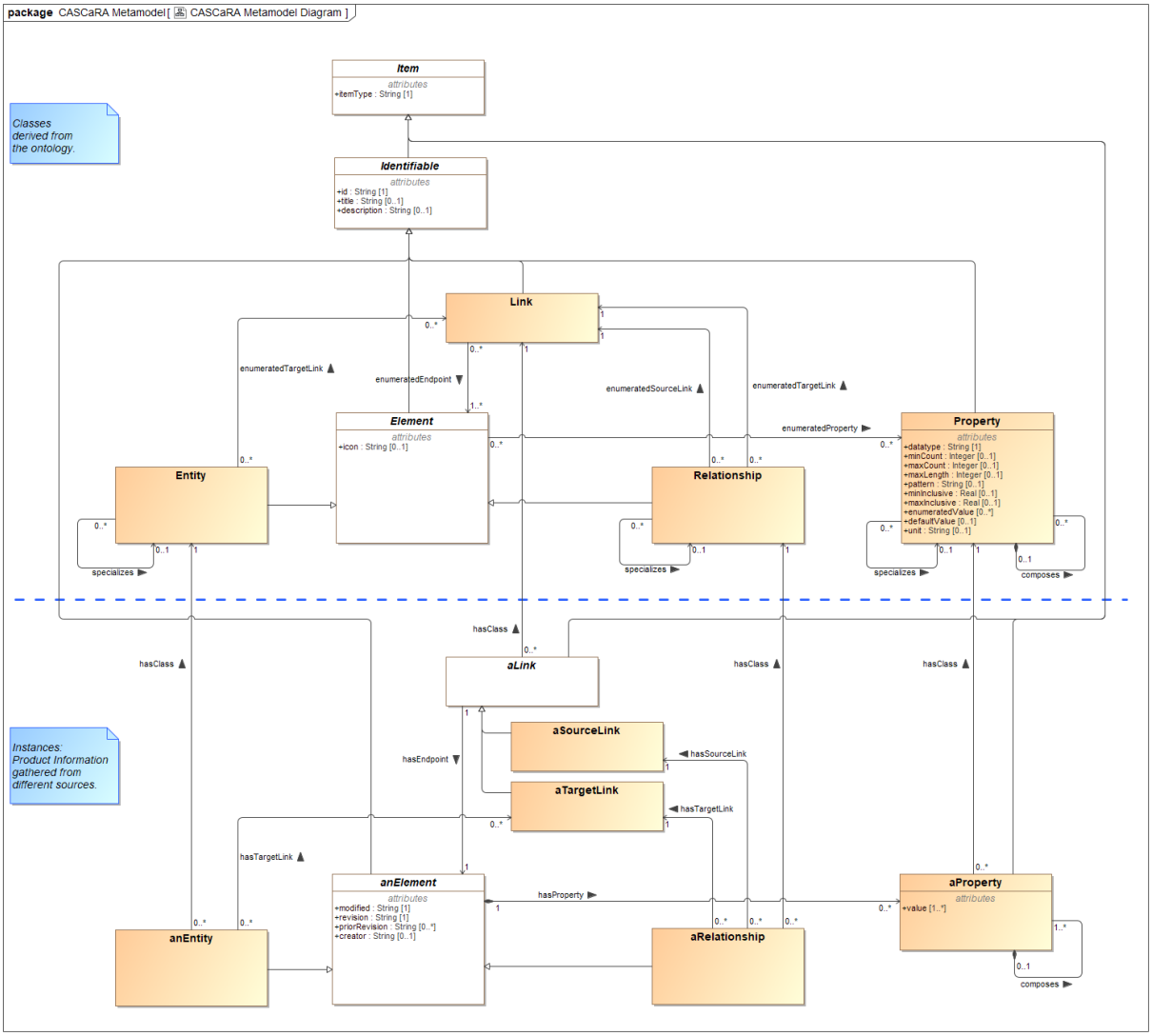
The metamodel is a 'platform independent model' in OMG terms and defines both the classes (above the dashed line) and their instances (below the dashed line) for the CASCaRA Product Information Graph. Both the classes and the instances are data:

- Entity and anEntity are used for model elements, such as a requirement, a function or a system component.
- Relationship and aRelationship are used for relations between elements, such as a 'system component satisfies a requirement'. A relationship class may define the classes of entities or relationships, whose instances are eligible as source or target in a directed relationship.
- Every Entity and Relationship can define an individual set of Property, each of which specifies a data type and range. For example, a requirement might have three properties, such as a „Title" with data type „String of max. length 96", a „Description" with data type „String" without restriction and a „Priority" with data type „String with a single choice of ['high', 'medium', 'low']".
- A datatype may be any of those known from XML. As described above, some parameters may restrict the value range, such as minimum and/or maximum value of a number, a string length or a set of enumerated values.
- anEntity or aRelationship is an instance of Entity or Relationship respectively. Each instance usually has a set of aProperty corresponding with the Property of its class.

Note:

- "Technical" super-class and sub-class refer to the UML elements used in this metamodel by means of UML:Generalization.

- "Semantic" super-class and sub-class refer to associations between the real world instances of the metamodel items as specified by the UML associations named 'specializes'.
- As usual, the classes with white background and italic name are abstract.



Properties

Element Type	Diagram
Notation	UML Class Diagram

Statements

CASCaRA Metamodel	*contains*	▣ CASCaRA Metamodel Diagram
▣ CASCaRA Metamodel Diagram	*shows*	☆ Link ☆ Identifiable o unit o enumeratedValue ☆ aLink o minInclusive o icon

	○ creator ○ title ☆ Item ☆ aRelationship ☆ Entity ○ minCount ☆ anElement ○ id ○ defaultValue ○ value ○ itemType ○ datatype ☆ Property ☆ Element ○ maxCount ○ description ○ maxInclusive ○ priorRevision ○ modified ☆ Relationship ○ pattern ☆ aTargetLink ☆ anEntity ☆ aProperty ○ maxLength ☆ aSourceLink ○ revision

8.2.2. ☆ Item

The technical super-class of all CASCaRA metamodel items specifying the type of item to allow for simple schema checking and transformation.
Details:

- The class is abstract.

Properties

Element Type	Class

Statements

CASCaRA Metamodel	*contains*	☆ Item
☆ Item	*has part*	○ itemType
☆ aProperty ☆ Identifiable ☆ aLink	*specializes*	☆ Item
▣ CASCaRA Metamodel Diagram	*shows*	☆ Item

8.2.3. ○ itemType

Here it is modeled as a string, but in fact it is an enumeration containing all concrete metamodel items (represented as UML classes), thus [Link, Property, Entity, Relationship, aPackage, aSourceLink, aTargetLink, aProperty, anEntity, aRelationship].

Properties

Element Type	uml:Property

Statements

☆ Item	*has part*	○ itemType
▣ CASCaRA Metamodel Diagram	*shows*	○ itemType

8.2.4. ☆ Identifiable

All technical sub-classes of this class are identifiable, means they have an identifier, a title (name) and optionally a description.
Details:

- The class is abstract.
- All attributes are inherited by its subclasses.
- All class level items in the upper part of the diagram are commonly released and made available with an URL path including the revision. Hence, multiple revisions of the class set may exist in parallel.
- All instance level items in the lower part of the diagram may have individual revisions with their respective revision, modification date and creator. The prior revision identifier is memorized to support change history and branching. In case of a merge, there are two prior revisions.

Properties

Element Type	Class

Statements

CASCaRA Metamodel	*contains*	☆ Identifiable
☆ Identifiable	*has part*	○ id ○ title ○ description
☆ Element ☆ Link ☆ anElement ☆ Property	*specializes*	☆ Identifiable
☆ Identifiable	*specializes*	☆ Item
▣ CASCaRA Metamodel Diagram	*shows*	☆ Identifiable

8.2.5. ○ id

Properties

Element Type	uml:Property

Statements

☆ Identifiable	*has part*	○ id
▣ CASCaRA Metamodel Diagram	*shows*	○ id

8.2.6. ○ title

Properties

Element Type	uml:Property

Statements

☆ Identifiable	*has part*	○ title
▣ CASCaRA Metamodel Diagram	*shows*	○ title

8.2.7. ○ description

Properties

Element Type	uml:Property

Statements

☆ Identifiable	*has part*	○ description
▣ CASCaRA Metamodel Diagram	*shows*	○ description

8.2.8. ☆ Element

A technical super-class of Entity (class of entities) and Relationship (class of relationships).
Details:

- The class is abstract.
- All attributes and associations are inherited by its technical sub-classes Entity and Relationship.

Properties

Element Type	Class

Statements

CASCaRA Metamodel	*contains*	☆ Element
☆ Element	*has part*	○ icon
☆ Element	*enumeratedProperty*	☆ Property
☆ Link	*enumeratedEndpoint*	☆ Element
☆ Entity ☆ Relationship	*specializes*	☆ Element
☆ Element	*specializes*	☆ Identifiable
▣ CASCaRA Metamodel Diagram	*shows*	☆ Element

8.2.9. ○ icon

An icon defined with a class can be used to decorate the instances.
Example:

- Folders as sub-class of Organizer and thus Entity may specify a UTF-8 character 🗀 for use by folders grouping certain model-elements.
- A data-URL of an icon is also permitted.

Properties

Element Type	uml:Property

Statements

☆ Element	*has part*	○ icon

▣ CASCaRA Metamodel Diagram	*shows*	○ icon

8.2.10. ☆ Property

Every class Entity and Relationship can have an individual set of configurable properties, each of which is uniquely defined by datatype and range. A Property is an owl:ObjectProperty when enumeratedValues are defined and is an owl:DatatypeProperty otherwise.
Example: A requirement might have a list with three properties, such as

- *Title* with data type „String of max. length 96“,
- *Description* with data type „String“ without length restriction and
- *Priority* with data type „String with a single choice of ['high', 'medium', 'low']“.

Details:

- Property class must have a title and may have a description named.
- Property class may define a minimum and maximum count of values.
- Property class must define a datatype and may restrict its range by minInclusive, maxInclusive or or a pattern defined by Regular Expression.
- Property class may define a set of enumerated values which must of course satisfy its own data type and range
- A property class may define a default value which must of course satisfy its own data type and range.
- Property class may be composed of multiple property classes to form a structured data type (xs:complexType). The structure must be a tree, thus without cyclic dependency.

Properties

Element Type	Class

Statements

CASCaRA Metamodel	*contains*	☆ Property
☆ Property	*has part*	○ datatype ○ minCount ○ maxCount ○ maxLength ○ pattern ○ minInclusive ○ maxInclusive ○ enumeratedValue ○ defaultValue ○ unit
☆ aProperty	*hasClass*	☆ Property
☆ Element	*enumeratedProperty*	☆ Property
☆ Property	*specializes*	☆ Property ☆ Identifiable
☆ Property	*composes*	☆ Property
▣ CASCaRA Metamodel Diagram	*shows*	☆ Property

8.2.11. ○ datatype

One of the xs: data types. Shall be anyURI.

Properties

Element Type	uml:Property

Statements

☆ Property	*has part*	○ datatype
▣ CASCaRA Metamodel Diagram	*shows*	○ datatype

8.2.12. ○ minCount

The minimal count an instance may have properties of this class. Is '0' by default.

Properties

Element Type	uml:Property

Statements

☆ Property	*has part*	○ minCount
▣ CASCaRA Metamodel Diagram	*shows*	○ minCount

8.2.13. ○ maxCount

The maximal count an instance may have properties of this class. Is unbounded (infinite) by default.

Properties

Element Type	uml:Property

Statements

☆ Property	*has part*	○ maxCount
▣ CASCaRA Metamodel Diagram	*shows*	○ maxCount

8.2.14. ○ maxLength

Maximum length of a property value with datatype 'xs:string'. Is not allowed with any other datatype.

8.2.14.1. Properties

Element Type	uml:Property

8.2.14.2. Statements

☆ Property	*has part*	○ maxLength
▣ CASCaRA Metamodel Diagram	*shows*	○ maxLength

8.2.15. ○ pattern

A Regular Expression that must evaluate successfully with the property value. Can also be used to limit the number of decimals of a number, as there is no restriction like xs:fractionDigits in SHACL.

Properties

Element Type	uml:Property

Statements

☆ Property	*has part*	○ pattern
▣ CASCaRA Metamodel Diagram	*shows*	○ pattern

8.2.16. ○ minInclusive

The minimum value a property of this type may have. May be applied only for numeric datatypes. It the intrinsic minimum value of the datatype by default.

Properties

Element Type	uml:Property	

Statements

☆ Property	*has part*	○ minInclusive
▣ CASCaRA Metamodel Diagram	*shows*	○ minInclusive

8.2.17. ○ maxInclusive

The maximum value a property of this type may have. May be applied only for numeric datatypes. It the intrinsic maximum value of the datatype by default.

Properties

Element Type	uml:Property

Statements

☆ Property	*has part*	○ maxInclusive
▣ CASCaRA Metamodel Diagram	*shows*	○ maxInclusive

8.2.18. ○ enumeratedValue

A property may be assigned zero to many values enumerated by its class. Enumerated values can be defined for all datatypes. By default all values in the range of the datatype can be assigned.

Properties

Element Type	uml:Property

Statements

☆ Property	*has part*	○ enumeratedValue
▣ CASCaRA Metamodel Diagram	*shows*	○ enumeratedValue

8.2.19. ○ defaultValue

Optional default value(s) in case a modelElement's property does not have an individual value. It must follow the definitions of the PropertyClass, of course.
Here, the attribute is named 'sh:defaultValue' for clarity and may just be called 'values' in an implementation, as all restrictions and operations of a property's values apply.

Properties

Element Type	uml:Property

Statements

☆ Property	*has part*	○ defaultValue
▣ CASCaRA Metamodel Diagram	*shows*	○ defaultValue

8.2.20. ○ unit

A unit as defined by the International System of Units (SI). A unit can only be defined for a numeric datatype. By default the property has no unit.

Properties

Element Type	uml:Property

Statements

☆ Property	*has part*	○ unit
▣ CASCaRA Metamodel Diagram	*shows*	○ unit

8.2.21. ☆ Link

A Link is an identifiable link class. A Link is an owl:ObjectProperty.
Example:

- An Organizer (semantic sub-class of Entity) may list Model-Elements (another semantic sub-class of Entity) to provide a document outline.
- As such the Link class *lists* is specified as enumeratedTargetLink by Organizer.

Properties

Element Type	Class

Statements

CASCaRA Metamodel	*contains*	☆ Link
☆ aLink	*hasClass*	☆ Link
☆ Entity ☆ Relationship	*enumeratedTargetLink*	☆ Link
☆ Link	*enumeratedEndpoint*	☆ Element
☆ Relationship	*enumeratedSourceLink*	☆ Link
☆ Link	*specializes*	☆ Identifiable
▣ CASCaRA Metamodel Diagram	*shows*	☆ Link

8.2.22. ☆ Entity

Entity is a class of model elements, such as a system component or a requirement. An Entity is an owl:Class.
Details:

- Entity should have a title (name) and may have a description (definition).
- Entity may specify (configure) zero to many classes of properties its instances may have.
- The definitions of the entity together with those of the properties may be used to build user dialogs with input verification as well as to check its instances whether all property values have a correct type and value range as well as whether all required ones are present.

Properties

Element Type	Class

Statements

CASCaRA Metamodel	*contains*	☆ Entity
☆ Entity	*specializes*	☆ Entity ☆ Element
☆ Entity	*enumeratedTargetLink*	☆ Link
☆ anEntity	*hasClass*	☆ Entity
▣ CASCaRA Metamodel Diagram	*shows*	☆ Entity

8.2.23. ☆ Relationship

Relationship is a class of relationships between two entities to allow assertions such as 'a system component satisfies a requirement'. All relationships are bilateral and directed. This allows for statements according to propositional logic. They can be easily mapped to many technologies such as RDF or even ReqIF. Being reified, a Relationship is an owl:Class.
Details:

- Relationship must have a title (name) and may have a description (definition).
- Relationship lists zero to many property classes its instances may have.
- Relationship may define (configure) entitiy classes or relationship classes, whose instances are eligible as source resp. target in a relationship. If none are defined, all entities and relationships are eligible.
- Relationship is reified (as a rdfs:Resource in RDF), so that it can be a source or target of another Relationship - to make a statement on a statement.
- The definitions of a relationship together with those of their properties may be used to build user dialogs with input verfication and to check its instances whether all properties have a correct type and value range as well as whether all required ones are present.

Properties

Element Type	Class

Statements

CASCaRA Metamodel	*contains*	☆ Relationship
☆ aRelationship	*hasClass*	☆ Relationship
☆ Relationship	*enumeratedTargetLink*	☆ Link
☆ Relationship	*specializes*	☆ Relationship ☆ Element
☆ Relationship	*enumeratedSourceLink*	☆ Link
▣ CASCaRA Metamodel Diagram	*shows*	☆ Relationship

8.2.24. ☆ anElement

A technical superclass of anEntity and aRelationship.
Details:

- The class is abstract.
- All attributes and associations are inherited by its sub-classes.
- Sub-classes may have individual revisions with revision id, modification date and creator. The prior revision id is memorized to support change history and branching. In case of a merge, there are two prior revisions.

Properties

Element Type	Class

Statements

CASCaRA Metamodel	*contains*	☆ anElement
☆ anElement	*has part*	o modified o revision o priorRevision o creator
☆ anElement	*hasProperty*	☆ aProperty
☆ aLink	*hasEndpoint*	☆ anElement
☆ anEntity ☆ aRelationship	*specializes*	☆ anElement
☆ anElement	*specializes*	☆ Identifiable
▣ CASCaRA Metamodel Diagram	*shows*	☆ anElement

8.2.25. ○ modified

A ISO-8601 DateTime value recording the point in time of the last change.

Properties

Element Type	uml:Property

Statements

☆ anElement	*has part*	○ modified
▣ CASCaRA Metamodel Diagram	*shows*	○ modified

8.2.26. ○ revision

A unique string identifying the revision of the entity or relationship instance.

Properties

Element Type	uml:Property

Statements

☆ anElement	*has part*	○ revision
▣ CASCaRA Metamodel Diagram	*shows*	○ revision

8.2.27. ○ priorRevision

Links to the previous revisions of an entity or relationship instance. It is a list with

- zero elements, if it is the first revision,
- one element for a subsequent revision or
- two elements when two branches are merged.

Properties

Element Type	uml:Property

Statements

☆ anElement	*has part*	○ priorRevision
▣ CASCaRA Metamodel Diagram	*shows*	○ priorRevision

8.2.28. ○ creator

The person committing the last change. Is often an e-mail address.

Properties

Element Type	uml:Property

Statements

☆ anElement	*has part*	○ creator
▣ CASCaRA Metamodel Diagram	*shows*	○ creator

8.2.29. ☆ aProperty

Each property belongs to a single Entity or Relationship. A property has *no* identifier, thus a property update results in a new revision of the element to which it belongs. Is 'payload' having a clear meaning through the respective Property class.
Details:

- A property may be required or not depending on the attributes sh:minCount and sh:maxCount of its class, but an existing property must have a value.
- A property may be composed of multiple properties to form a structured data set according to its Property class.

Properties

Element Type	Class

Statements

CASCaRA Metamodel	*contains*	☆ aProperty
☆ aProperty	*has part*	○ value
☆ anElement	*hasProperty*	☆ aProperty
☆ aProperty	*composes*	☆ aProperty
☆ aProperty	*hasClass*	☆ Property
☆ aProperty	*specializes*	☆ Item
▣ CASCaRA Metamodel Diagram	*shows*	☆ aProperty

8.2.30. ○ value

The value of a property.
Details:

- If the datatype of the property's class is string, the value is a language string consisting of value and IETF language tag.
- Otherwise the value is a simple value.

Properties

Element Type	uml:Property

Statements

☆ aProperty	*has part*	○ value
▣ CASCaRA Metamodel Diagram	*shows*	○ value

8.2.31. ☆ aLink

Is a technical super-class for link instances.
Details:

- The class is abstract.
- A list of enumerated endpoints (references) per link instance is possible.

Properties

Element Type	Class

Statements

CASCaRA Metamodel	*contains*	☆ aLink
☆ aLink	*hasClass*	☆ Link
☆ aLink	*hasEndpoint*	☆ anElement
☆ aSourceLink ☆ aTargetLink	*specializes*	☆ aLink
☆ aLink	*specializes*	☆ Item
▣ CASCaRA Metamodel Diagram	*shows*	☆ aLink

8.2.32. ☆ aSourceLink

A link instance serving as source for a reified relationship.

From a semantic point of view, there is no need to make a distinction between aSourceLink and aTargetLink at the metamodel level. The distinction can be done and *is* made at ontology level in the CASCaRA Semantic Infrastructure. However it is much easier to check the pattern/schema/shape, if the metamodel item type is explicit.
Example:

- A single 'satisfies' relationship can relate one or many function instances to a requirement instance.

Properties

Element Type	Class

Statements

CASCaRA Metamodel	*contains*	☆ aSourceLink
☆ aRelationship	*hasSourceLink*	☆ aSourceLink
☆ aSourceLink	*specializes*	☆ aLink
▣ CASCaRA Metamodel Diagram	*shows*	☆ aSourceLink

8.2.33. ☆ aTargetLink

A link instance serving as target for a reified relationship.
From a semantic point of view, there is no need to make a distinction between aSourceLink and aTargetLink at the metamodel level. The distinction can be done and *is* made at ontology level in the CASCaRA Semantic Infrastructure. However it is much easier to check the pattern/schema/shape, if the metamodel item type is explicit.
Example:

- A single 'satisfies' relationship can relate a function instance to one or multiple requirement instances (targets).

Properties

Element Type	Class

Statements

CASCaRA Metamodel	*contains*	☆ aTargetLink
☆ anEntity ☆ aRelationship	*hasTargetLink*	☆ aTargetLink
☆ aTargetLink	*specializes*	☆ aLink
▣ CASCaRA Metamodel Diagram	*shows*	☆ aTargetLink

8.2.34. ☆ anEntity

Is used for instances of model-elements such as a system component or a requirement. Is 'payload' having a clear meaning through the respective Entity class.
Details:

- An entity must have either a title or a description - or both. This allows to compose documents including text paragraphs without title.
- The entity's class enumerates the classes of properties it may have.

Properties

Element Type	Class

Statements

CASCaRA Metamodel	*contains*	☆ anEntity
☆ anEntity	*hasTargetLink*	☆ aTargetLink

☆ anEntity	*hasClass*	☆ Entity
☆ anEntity	*specializes*	☆ anElement
▣ CASCaRA Metamodel Diagram	*shows*	☆ anEntity

8.2.35. ☆ aRelationship

Is used for relations between entities, such as a 'system component satisfies a requirement'. All relationships are bilateral and directed. Is 'payload' having a clear meaning through the respective Relationship class.
Details:

- This is a reified graph edge and must have exactly one source and target entity each. The source must be an entity or relationship with a class defined by an eligible source class of its Relationship. The same applies for the target.
- Even though a relationship instance may have an individual title, it is advised to omit it and assume its class' title when needed to assure that relationships of the same class are named equally.
- The relationship's class enumerates the classes of properties it may have.

Properties

Element Type	Class

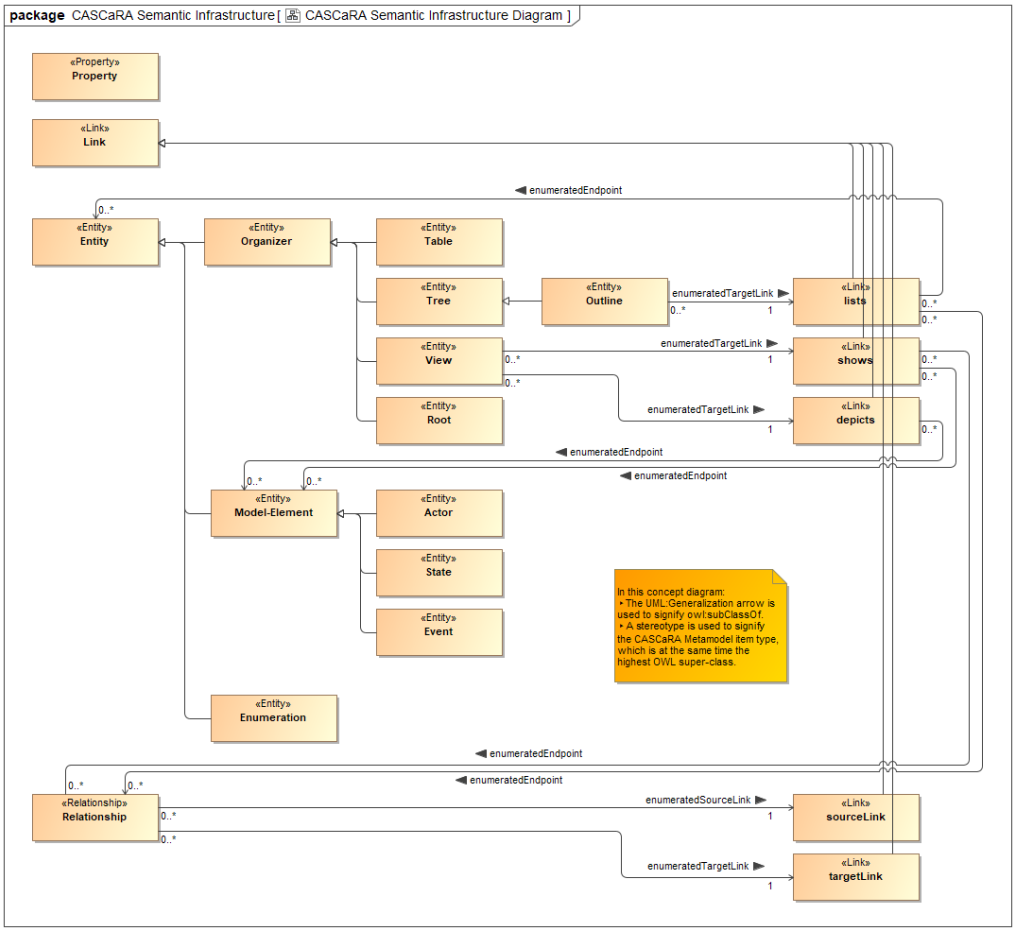
Statements

CASCaRA Metamodel	*contains*	☆ aRelationship
☆ aRelationship	*hasClass*	☆ Relationship
☆ aRelationship	*hasTargetLink*	☆ aTargetLink
☆ aRelationship	*hasSourceLink*	☆ aSourceLink
☆ aRelationship	*specializes*	☆ anElement
▣ CASCaRA Metamodel Diagram	*shows*	☆ aRelationship

8.3. CASCaRA Semantic Infrastructure

8.3.1. ▣ CASCaRA Semantic Infrastructure Diagram

The CASCaRA Semantic Infrastructure complements the CASCaRA Metamodel by further sub-classing its items. The idea is to keep the metamodel as simple as possible without sacrificing semantic precision.
Let's consider the paradigma of separating syntax and semantics with the objective to shield the software from an ontology under evolution over time. In all likelihood, the CASCaRA Semantic Infrastructure will be used by software for reasons of simplicity. Therefore it is advised to treat the semantic infrastructure similarly to the metamodel: Change it seldomly and keep it downward compatible. That is, software built assuming an older version shall not break when a new version is adopted).
In any case an ontology for use with the CASCaRA Metamodel SHALL sub-class the items of the CASCaRA Semantic Infrastructure.



Properties

Element Type	Diagram
Notation	UML Class Diagram

Statements

CASCaRA Semantic Infrastructure	*contains*	▣ CASCaRA Semantic Infrastructure Diagram
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ Table ☆ targetLink ☆ Link ☆ sourceLink ☆ Root ☆ Actor ☆ View ☆ depicts ☆ lists ☆ Outline ☆ State ☆ Enumeration ☆ Relationship

		☆ Property
		☆ shows
		☆ Event
		☆ Organizer
		☆ Entity
		☆ Model-Element
		☆ Tree

8.3.2. ☆ Property

The semantic foundation for all Property (class) definitions. Is an owl:DatatypeProperty when the value is a literal or an owl:ObjectProperty when the value is a reference to an enumerated value.

Properties

Element Type	cas_meta:Property

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ Property
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ Property

8.3.3. ☆ Link

The semantic foundation for all Link (class) definitions. Is an owl:ObjectProperty.

Properties

Element Type	cas_meta:Link

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ Link
☆ depicts ☆ lists ☆ shows ☆ targetLink ☆ sourceLink	*specializes*	☆ Link
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ Link

8.3.4. ☆ lists

A Link class for use by Outline to enumerate eligible Entity and/or Relationship classes.

Properties

Element Type	cas_meta:Link

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ lists
☆ lists	*enumeratedEndpoint*	☆ Relationship ☆ Entity
☆ Outline	*enumeratedTargetLink*	☆ lists
☆ lists	*specializes*	☆ Link
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ lists

8.3.5. ☆ shows

A Link class for use by View to enumerate eligible Entity and/or Relationship classes that are visible on a view (e.g. model-diagram).

Properties

Element Type	cas_meta:Link

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ shows
☆ shows	*enumeratedEndpoint*	☆ Model-Element
☆ View	*enumeratedTargetLink*	☆ shows
☆ shows	*specializes*	☆ Link
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ shows

8.3.6. ☆ depicts

A Link class for use by View to enumerate eligible Entity and/or Relationship classes that can own a view (e.g. model-diagram).

Properties

Element Type	cas_meta:Link

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ depicts
☆ depicts	*enumeratedEndpoint*	☆ Model-Element
☆ View	*enumeratedTargetLink*	☆ depicts
☆ depicts	*specializes*	☆ Link
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ depicts

8.3.7. ☆ sourceLink

A Link class for use by View to enumerate eligible Entity and/or Relationship classes that can serve as a source of a Relationship.
By further specializing this class, a speaking title can be given to a sourceLink, such as 'employee' in a 'worksFor' relationship being a specialization of Relationship.

Properties

Element Type	cas_meta:Link

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ sourceLink
☆ Relationship	*enumeratedSourceLink*	☆ sourceLink
☆ sourceLink	*specializes*	☆ Link
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ sourceLink

8.3.8. ☆ targetLink

A Link class for use by View to enumerate eligible Entity and/or Relationship classes that can serve as a target of a Relationship.
By further specializing this class, a speaking title can be given to a targetLink, such as 'employer' in a 'worksFor' relationship being a specialization of Relationship.

Properties

Element Type	cas_meta:Link	

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ targetLink
☆ Relationship	*enumeratedTargetLink*	☆ targetLink
☆ targetLink	*specializes*	☆ Link
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ targetLink

8.3.9. ☆ Entity

The semantic foundation for all Entity (class) definitions. Is an owl:Class.

Properties

Element Type	cas_meta:Entity

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ Entity
☆ lists	*enumeratedEndpoint*	☆ Entity
☆ Enumeration ☆ Model-Element ☆ Organizer	*specializes*	☆ Entity
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ Entity

8.3.10. ☆ Organizer

An Organizer is about *presenting* genuine Model-Elements with a target group of users in mind, for example a document outline, a table or a diagram.

Properties

Element Type	cas_meta:Entity

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ Organizer
☆ Table ☆ View ☆ Tree ☆ Root	*specializes*	☆ Organizer
☆ Organizer	*specializes*	☆ Entity
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ Organizer

8.3.11. ☆ Table

A specialized Organizer with elements (cells) arranged two-dimensionally in columns and rows. Any element SHALL contain a reference to any Entity or Relationship.

Properties

Element Type	cas_meta:Entity

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ Table
☆ Table	*specializes*	☆ Organizer

▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ Table

8.3.12. ☆ Tree

A specialized Organizer with elements (nodes) arranged in a strictly hierarchical manner. Any element SHALL contain a reference to any Entity or Relationship.

Properties

Element Type	cas_meta:Entity

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ Tree
☆ Outline	*specializes*	☆ Tree
☆ Tree	*specializes*	☆ Organizer
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ Tree

8.3.13. ☆ Outline

A tree (hierarchical arrangement) of references to graph nodes (vertices) to create a reading sequence. Graph nodes include Entity and Relationship. Outline is used at any level of the tree like a directory folder or universally for section, chapter and paragraph of a document.

Properties

Element Type	cas_meta:Entity

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ Outline
☆ Outline	*enumeratedTargetLink*	☆ lists
☆ Outline	*specializes*	☆ Tree
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ Outline

8.3.14. ☆ View

A specialized Organizer with elements displayed in two or three dimensions such as a model-diagram, 3D visualization or pie-chart. Any element SHOULD reference any Entity or Relationship.

Properties

Element Type	cas_meta:Entity

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ View
☆ View	*enumeratedTargetLink*	☆ shows ☆ depicts
☆ View	*specializes*	☆ Organizer
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ View

8.3.15. ☆ Root

A root class serves to anchor organizer structures in a data package. While a graph has no start and no end, a root instance with a subordinated organizer structure offers a way to explore the graph for a given purpose.
Example:

- A document outline anchored at a root element would define a document to read by auditors and perhaps another one for potential customers.

- When querying all instances of class Root, a list of available documents is returned.

Properties

Element Type	cas_meta:Entity

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ Root
☆ Root	*specializes*	☆ Organizer
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ Root

8.3.16. ☆ Model-Element

A Model-Element is an artifact *representing* a genuine part of a system specification in contrast to an Organizer which is about *presenting* model-elements.

Properties

Element Type	cas_meta:Entity

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ Model-Element
☆ depicts ☆ shows	*enumeratedEndpoint*	☆ Model-Element
☆ Actor ☆ Event ☆ State	*specializes*	☆ Model-Element
☆ Model-Element	*specializes*	☆ Entity
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ Model-Element

8.3.17. ☆ Actor

A fundamental model-element class for actors (e.g. users, functions, systems, components, ...).

Properties

Element Type	cas_meta:Entity

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ Actor
☆ Actor	*specializes*	☆ Model-Element
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ Actor

8.3.18. ☆ State

A fundamental model-element class for states (e.g. system or process states, information, form, color, ...).

Properties

Element Type	cas_meta:Entity

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ State
☆ State	*specializes*	☆ Model-Element
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ State

8.3.19. ☆ Event

A fundamental model-element class for events (e.g. environmental or process events, ...).

Properties

Element Type	cas_meta:Entity

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ Event
☆ Event	*specializes*	☆ Model-Element
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ Event

8.3.20. ☆ Enumeration

A finite list of values for Properties.

Properties

Element Type	cas_meta:Entity

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ Enumeration
☆ Enumeration	*specializes*	☆ Entity
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ Enumeration

8.3.21. ☆ Relationship

The semantic foundation for all Relationship (class) definitions. Is an owl:Class.

Properties

Element Type	cas_meta:Relationship

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ Relationship
☆ lists	*enumeratedEndpoint*	☆ Relationship
☆ Relationship	*enumeratedTargetLink*	☆ targetLink
☆ Relationship	*enumeratedSourceLink*	☆ sourceLink
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ Relationship

8.4. Model Elements (Glossary)

Properties

Element Type	Model Elements (Glossary)

8.4.1. ☆ Actor

A fundamental model-element class for actors (e.g. users, functions, systems, components, ...).

Properties

Element Type	cas_meta:Entity

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ Actor
☆ Actor	*specializes*	☆ Model-Element
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ Actor

8.4.2. ☆ aLink

Is a technical super-class for link instances.
Details:

- The class is abstract.
- A list of enumerated endpoints (references) per link instance is possible.

Properties

Element Type	Class

Statements

CASCaRA Metamodel	*contains*	☆ aLink
☆ aLink	*hasClass*	☆ Link
☆ aLink	*hasEndpoint*	☆ anElement
☆ aSourceLink ☆ aTargetLink	*specializes*	☆ aLink
☆ aLink	*specializes*	☆ Item
▣ CASCaRA Metamodel Diagram	*shows*	☆ aLink

8.4.3. ☆ anElement

A technical superclass of anEntity and aRelationship.
Details:

- The class is abstract.
- All attributes and associations are inherited by its sub-classes.
- Sub-classes may have individual revisions with revision id, modification date and creator. The prior revision id is memorized to support change history and branching. In case of a merge, there are two prior revisions.

Properties

Element Type	Class

Statements

CASCaRA Metamodel	*contains*	☆ anElement
☆ anElement	*has part*	o modified o revision o priorRevision o creator
☆ anElement	*hasProperty*	☆ aProperty
☆ aLink	*hasEndpoint*	☆ anElement
☆ anEntity ☆ aRelationship	*specializes*	☆ anElement
☆ anElement	*specializes*	☆ Identifiable
▣ CASCaRA Metamodel Diagram	*shows*	☆ anElement

8.4.4. ☆ anEntity

Is used for instances of model-elements such as a system component or a requirement. Is 'payload' having a clear meaning through the respective Entity class.
Details:

- An entity must have either a title or a description - or both. This allows to compose documents including text paragraphs without title.
- The entity's class enumerates the classes of properties it may have.

Properties

Element Type	Class

Statements

CASCaRA Metamodel	*contains*	☆ anEntity
☆ anEntity	*hasTargetLink*	☆ aTargetLink
☆ anEntity	*hasClass*	☆ Entity
☆ anEntity	*specializes*	☆ anElement
▣ CASCaRA Metamodel Diagram	*shows*	☆ anEntity

8.4.5. ☆ aProperty

Each property belongs to a single Entity or Relationship. A property has *no* identifier, thus a property update results in a new revision of the element to which it belongs. Is 'payload' having a clear meaning through the respective Property class.
Details:

- A property may be required or not depending on the attributes sh:minCount and sh:maxCount of its class, but an existing property must have a value.
- A property may be composed of multiple properties to form a structured data set according to its Property class.

Properties

Element Type	Class

Statements

CASCaRA Metamodel	*contains*	☆ aProperty
☆ aProperty	*has part*	○ value
☆ anElement	*hasProperty*	☆ aProperty
☆ aProperty	*composes*	☆ aProperty
☆ aProperty	*hasClass*	☆ Property
☆ aProperty	*specializes*	☆ Item
▣ CASCaRA Metamodel Diagram	*shows*	☆ aProperty

8.4.6. ☆ aRelationship

Is used for relations between entities, such as a 'system component satisfies a requirement'. All relationships are bilateral and directed. Is 'payload' having a clear meaning through the respective Relationship class.
Details:

- This is a reified graph edge and must have exactly one source and target entity each. The source must be an entity or relationship with a class defined by an eligible source class of its Relationship. The same applies for the target.
- Even though a relationship instance may have an individual title, it is advised to omit it and assume its class' title when needed to assure that relationships of the same class are named equally.
- The relationship's class enumerates the classes of properties it may have.

Properties

Element Type	Class	

Statements

CASCaRA Metamodel	*contains*	☆ aRelationship
☆ aRelationship	*hasClass*	☆ Relationship
☆ aRelationship	*hasTargetLink*	☆ aTargetLink
☆ aRelationship	*hasSourceLink*	☆ aSourceLink
☆ aRelationship	*specializes*	☆ anElement
▣ CASCaRA Metamodel Diagram	*shows*	☆ aRelationship

8.4.7. ☆ aSourceLink

A link instance serving as source for a reified relationship.
From a semantic point of view, there is no need to make a distinction between aSourceLink and aTargetLink at the metamodel level. The distinction can be done and *is* made at ontology level in the CASCaRA Semantic Infrastructure. However it is much easier to check the pattern/schema/shape, if the metamodel item type is explicit.
Example:

- A single 'satisfies' relationship can relate one or many function instances to a requirement instance.

Properties

Element Type	Class

Statements

CASCaRA Metamodel	*contains*	☆ aSourceLink
☆ aRelationship	*hasSourceLink*	☆ aSourceLink
☆ aSourceLink	*specializes*	☆ aLink
▣ CASCaRA Metamodel Diagram	*shows*	☆ aSourceLink

8.4.8. ☆ aTargetLink

A link instance serving as target for a reified relationship.
From a semantic point of view, there is no need to make a distinction between aSourceLink and aTargetLink at the metamodel level. The distinction can be done and *is* made at ontology level in the CASCaRA Semantic Infrastructure. However it is much easier to check the pattern/schema/shape, if the metamodel item type is explicit.
Example:

- A single 'satisfies' relationship can relate a function instance to one or multiple requirement instances (targets).

Properties

Element Type	Class

Statements

CASCaRA Metamodel	*contains*	☆ aTargetLink
☆ anEntity ☆ aRelationship	*hasTargetLink*	☆ aTargetLink
☆ aTargetLink	*specializes*	☆ aLink
▣ CASCaRA Metamodel Diagram	*shows*	☆ aTargetLink

8.4.9. ○ creator

The person committing the last change. Is often an e-mail address.

Properties

Element Type	uml:Property

Statements

☆ anElement	*has part*	○ creator
▣ CASCaRA Metamodel Diagram	*shows*	○ creator

○ **datatype**

One of the xs: data types. Shall be anyURI.

Properties

Element Type	uml:Property

Statements

☆ Property	*has part*	○ datatype
▣ CASCaRA Metamodel Diagram	*shows*	○ datatype

8.4.10. ○ defaultValue

Optional default value(s) in case a modelElement's property does not have an individual value. It must follow the definitions of the PropertyClass, of course.
Here, the attribute is named 'sh:defaultValue' for clarity and may just be called 'values' in an implementation, as all restrictions and operations of a property's values apply.

Properties

Element Type	uml:Property

Statements

☆ Property	*has part*	○ defaultValue
▣ CASCaRA Metamodel Diagram	*shows*	○ defaultValue

8.4.11. ☆ depicts

A Link class for use by View to enumerate eligible Entity and/or Relationship classes that can own a view (e.g. model-diagram).

Properties

Element Type	cas_meta:Link

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ depicts
☆ depicts	*enumeratedEndpoint*	☆ Model-Element
☆ View	*enumeratedTargetLink*	☆ depicts
☆ depicts	*specializes*	☆ Link
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ depicts

8.4.12. ○ description

Properties

Element Type	uml:Property

Statements

☆ Identifiable	*has part*	○ description
▣ CASCaRA Metamodel Diagram	*shows*	○ description

8.4.13. ☆ Element

A technical super-class of Entity (class of entities) and Relationship (class of relationships).
Details:

- The class is abstract.
- All attributes and associations are inherited by its technical sub-classes Entity and Relationship.

Properties

Element Type	Class

Statements

CASCaRA Metamodel	*contains*	☆ Element
☆ Element	*has part*	○ icon
☆ Element	*enumeratedProperty*	☆ Property
☆ Link	*enumeratedEndpoint*	☆ Element
☆ Entity ☆ Relationship	*specializes*	☆ Element
☆ Element	*specializes*	☆ Identifiable
▣ CASCaRA Metamodel Diagram	*shows*	☆ Element

8.4.14. ☆ Entity

Entity is a class of model elements, such as a system component or a requirement. An Entity is an owl:Class.
Details:

- Entity should have a title (name) and may have a description (definition).
- Entity may specify (configure) zero to many classes of properties its instances may have.
- The definitions of the entity together with those of the properties may be used to build user dialogs with input verification as well as to check its instances whether all property values have a correct type and value range as well as whether all required ones are present.

8.4.14.1. Properties

Element Type	Class

8.4.14.2. Statements

CASCaRA Metamodel	*contains*	☆ Entity
☆ Entity	*specializes*	☆ Entity ☆ Element
☆ Entity	*enumeratedTargetLink*	☆ Link
☆ anEntity	*hasClass*	☆ Entity
▣ CASCaRA Metamodel Diagram	*shows*	☆ Entity

8.4.15. ☆ Entity

The semantic foundation for all Entity (class) definitions. Is an owl:Class.

Properties

Element Type	cas_meta:Entity

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ Entity
☆ lists	*enumeratedEndpoint*	☆ Entity
☆ Enumeration ☆ Model-Element ☆ Organizer	*specializes*	☆ Entity
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ Entity

8.4.16. ○ enumeratedValue

A property may be assigned zero to many values enumerated by its class. Enumerated values can be defined for all datatypes. By default all values in the range of the datatype can be assigned.

8.4.16.1. Properties

Element Type	uml:Property

8.4.16.2. Statements

☆ Property	*has part*	○ enumeratedValue
▣ CASCaRA Metamodel Diagram	*shows*	○ enumeratedValue

8.4.17. ☆ Enumeration

A finite list of values for Properties.

Properties

Element Type	cas_meta:Entity

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ Enumeration
☆ Enumeration	*specializes*	☆ Entity
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ Enumeration

8.4.18. ☆ Event

A fundamental model-element class for events (e.g. environmental or process events, ...).

Properties

Element Type	cas_meta:Entity

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ Event
☆ Event	*specializes*	☆ Model-Element
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ Event

8.4.19. ○ icon

An icon defined with a class can be used to decorate the instances.
Example:

- Folders as sub-class of Organizer and thus Entity may specify a UTF-8 character 🗀 for use by folders grouping certain model-elements.
- A data-URL of an icon is also permitted.

Properties

Element Type	uml:Property	

Statements

☆ Element	*has part*	○ icon
▣ CASCaRA Metamodel Diagram	*shows*	○ icon

8.4.20. ○ id

Properties

Element Type	uml:Property

Statements

☆ Identifiable	*has part*	○ id
▣ CASCaRA Metamodel Diagram	*shows*	○ id

8.4.21. ☆ Identifiable

All technical sub-classes of this class are identifiable, means they have an identifier, a title (name) and optionally a description.
Details:

- The class is abstract.
- All attributes are inherited by its subclasses.
- All class level items in the upper part of the diagram are commonly released and made available with an URL path including the revision. Hence, multiple revisions of the class set may exist in parallel.
- All instance level items in the lower part of the diagram may have individual revisions with their respective revision, modification date and creator. The prior revision identifier is memorized to support change history and branching. In case of a merge, there are two prior revisions.

Properties

Element Type	Class

Statements

CASCaRA Metamodel	*contains*	☆ Identifiable
☆ Identifiable	*has part*	○ id ○ title ○ description
☆ Element ☆ Link ☆ anElement ☆ Property	*specializes*	☆ Identifiable
☆ Identifiable	*specializes*	☆ Item
▣ CASCaRA Metamodel Diagram	*shows*	☆ Identifiable

8.4.22. ☆ Item

The technical super-class of all CASCaRA metamodel items specifying the type of item to allow for simple schema checking and transformation.
Details:

- The class is abstract.

Properties

Element Type	Class

Statements

CASCaRA Metamodel	*contains*	☆ Item
☆ Item	*has part*	○ itemType
☆ aProperty ☆ Identifiable ☆ aLink	*specializes*	☆ Item
▣ CASCaRA Metamodel Diagram	*shows*	☆ Item

8.4.23. ○ itemType

Here it is modeled as a string, but in fact it is an enumeration containing all concrete metamodel items (represented as UML classes), thus [Link, Property, Entity, Relationship, aPackage, aSourceLink, aTargetLink, aProperty, anEntity, aRelationship].

Properties

Element Type	uml:Property

Statements

☆ Item	*has part*	○ itemType
▣ CASCaRA Metamodel Diagram	*shows*	○ itemType

8.4.24. ☆ Link

A Link is an identifiable link class. A Link is an owl:ObjectProperty.
Example:

- An Organizer (semantic sub-class of Entity) may list Model-Elements (another semantic sub-class of Entity) to provide a document outline.
- As such the Link class *lists* is specified as enumeratedTargetLink by Organizer.

Properties

Element Type	Class

Statements

CASCaRA Metamodel	*contains*	☆ Link
☆ aLink	*hasClass*	☆ Link
☆ Entity ☆ Relationship	*enumeratedTargetLink*	☆ Link
☆ Link	*enumeratedEndpoint*	☆ Element
☆ Relationship	*enumeratedSourceLink*	☆ Link
☆ Link	*specializes*	☆ Identifiable
▣ CASCaRA Metamodel Diagram	*shows*	☆ Link

8.4.25. ☆ Link

The semantic foundation for all Link (class) definitions. Is an owl:ObjectProperty.

Properties

Element Type	cas_meta:Link

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ Link
☆ depicts		
☆ lists		
☆ shows	*specializes*	☆ Link
☆ targetLink		
☆ sourceLink		
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ Link

8.4.26. ☆ lists

A Link class for use by Outline to enumerate eligible Entity and/or Relationship classes.

Properties

Element Type	cas_meta:Link

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ lists
☆ lists	*enumeratedEndpoint*	☆ Relationship ☆ Entity
☆ Outline	*enumeratedTargetLink*	☆ lists
☆ lists	*specializes*	☆ Link
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ lists

8.4.27. ○ maxCount

The maximal count an instance may have properties of this class. Is unbounded (infinite) by default.

Properties

Element Type	uml:Property

Statements

☆ Property	*has part*	○ maxCount
▣ CASCaRA Metamodel Diagram	*shows*	○ maxCount

8.4.28. ○ maxInclusive

The maximum value a property of this type may have. May be applied only for numeric datatypes. It the intrinsic maximum value of the datatype by default.

Properties

Element Type	uml:Property

Statements

☆ Property	*has part*	○ maxInclusive
▣ CASCaRA Metamodel Diagram	*shows*	○ maxInclusive

8.4.29. ○ maxLength

Maximum length of a property value with datatype 'xs:string'. Is not allowed with any other datatype.

Properties

Element Type	uml:Property

Statements

☆ Property	*has part*	○ maxLength
▣ CASCaRA Metamodel Diagram	*shows*	○ maxLength

8.4.30. ○ minCount

The minimal count an instance may have properties of this class. Is '0' by default.

Properties

Element Type	uml:Property

Statements

☆ Property	*has part*	○ minCount
▣ CASCaRA Metamodel Diagram	*shows*	○ minCount

8.4.31. ○ minInclusive

The minimum value a property of this type may have. May be applied only for numeric datatypes. It the intrinsic minimum value of the datatype by default.

Properties

Element Type	uml:Property

Statements

☆ Property	*has part*	○ minInclusive
▣ CASCaRA Metamodel Diagram	*shows*	○ minInclusive

8.4.32. ☆ Model-Element

A Model-Element is an artifact *representing* a genuine part of a system specification in contrast to an Organizer which is about *presenting* model-elements.

Properties

Element Type	cas_meta:Entity

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ Model-Element
☆ depicts ☆ shows	*enumeratedEndpoint*	☆ Model-Element
☆ Actor ☆ Event ☆ State	*specializes*	☆ Model-Element
☆ Model-Element	*specializes*	☆ Entity
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ Model-Element

8.4.33. ○ modified

A ISO-8601 DateTime value recording the point in time of the last change.

Properties

Element Type	uml:Property

Statements

☆ anElement	*has part*	○ modified

▣ CASCaRA Metamodel Diagram	*shows*	○ modified

8.4.34. ☆ Organizer

An Organizer is about *presenting* genuine Model-Elements with a target group of users in mind, for example a document outline, a table or a diagram.

Properties

Element Type	cas_meta:Entity

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ Organizer
☆ Table ☆ View ☆ Tree ☆ Root	*specializes*	☆ Organizer
☆ Organizer	*specializes*	☆ Entity
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ Organizer

8.4.35. ☆ Outline

A tree (hierarchical arrangement) of references to graph nodes (vertices) to create a reading sequence. Graph nodes include Entity and Relationship. Outline is used at any level of the tree like a directory folder or universally for section, chapter and paragraph of a document.

Properties

Element Type	cas_meta:Entity

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ Outline
☆ Outline	*enumeratedTargetLink*	☆ lists
☆ Outline	*specializes*	☆ Tree
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ Outline

8.4.36. ○ pattern

A Regular Expression that must evaluate successfully with the property value. Can also be used to limit the number of decimals of a number, as there is no restriction like xs:fractionDigits in SHACL.

Properties

Element Type	uml:Property

Statements

☆ Property	*has part*	○ pattern
▣ CASCaRA Metamodel Diagram	*shows*	○ pattern

8.4.37. ○ priorRevision

Links to the previous revisions of an entity or relationship instance. It is a list with

- zero elements, if it is the first revision,
- one element for a subsequent revision or
- two elements when two branches are merged.

Properties

Element Type	uml:Property	

Statements

☆ anElement	*has part*	○ priorRevision
▣ CASCaRA Metamodel Diagram	*shows*	○ priorRevision

8.4.38. ☆ Property

Every class Entity and Relationship can have an individual set of configurable properties, each of which is uniquely defined by datatype and range. A Property is an owl:ObjectProperty when enumeratedValues are defined and is an owl:DatatypeProperty otherwise.
Example: A requirement might have a list with three properties, such as

- *Title* with data type „String of max. length 96“,
- *Description* with data type „String“ without length restriction and
- *Priority* with data type „String with a single choice of ['high', 'medium', 'low']“.

Details:

- Property class must have a title and may have a description named.
- Property class may define a minimum and maximum count of values.
- Property class must define a datatype and may restrict its range by minInclusive, maxInclusive or or a pattern defined by Regular Expression.
- Property class may define a set of enumerated values which must of course satisfy its own data type and range
- A property class may define a default value which must of course satisfy its own data type and range.
- Property class may be composed of multiple property classes to form a structured data type (xs:complexType). The structure must be a tree, thus without cyclic dependency.

Properties

Element Type	Class	

Statements

CASCaRA Metamodel	*contains*	☆ Property
☆ Property	*has part*	○ datatype ○ minCount ○ maxCount ○ maxLength ○ pattern ○ minInclusive ○ maxInclusive ○ enumeratedValue ○ defaultValue ○ unit
☆ aProperty	*hasClass*	☆ Property
☆ Element	*enumeratedProperty*	☆ Property
☆ Property	*specializes*	☆ Property ☆ Identifiable
☆ Property	*composes*	☆ Property
▣ CASCaRA Metamodel Diagram	*shows*	☆ Property

8.4.39. ☆ Property

The semantic foundation for all Property (class) definitions. Is an owl:DatatypeProperty when the value is a literal or an owl:ObjectProperty when the value is a reference to an enumerated value.

Properties

Element Type	cas_meta:Property

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ Property
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ Property

8.4.40. ☆ Relationship

Relationship is a class of relationships between two entities to allow assertions such as 'a system component satisfies a requirement'. All relationships are bilateral and directed. This allows for statements according to propositional logic. They can be easily mapped to many technologies such as RDF or even ReqIF. Being reified, a Relationship is an owl:Class.
Details:

- Relationship must have a title (name) and may have a description (definition).
- Relationship lists zero to many property classes its instances may have.
- Relationship may define (configure) entitiy classes or relationship classes, whose instances are eligible as source resp. target in a relationship. If none are defined, all entities and relationships are eligible.
- Relationship is reified (as a rdfs:Resource in RDF), so that it can be a source or target of another Relationship - to make a statement on a statement.
- The definitions of a relationship together with those of their properties may be used to build user dialogs with input verfication and to check its instances whether all properties have a correct type and value range as well as whether all required ones are present.

Properties

Element Type	Class

Statements

CASCaRA Metamodel	*contains*	☆ Relationship
☆ aRelationship	*hasClass*	☆ Relationship
☆ Relationship	*enumeratedTargetLink*	☆ Link
☆ Relationship	*specializes*	☆ Relationship ☆ Element
☆ Relationship	*enumeratedSourceLink*	☆ Link
▣ CASCaRA Metamodel Diagram	*shows*	☆ Relationship

8.4.41. ☆ Relationship

The semantic foundation for all Relationship (class) definitions. Is an owl:Class.

Properties

Element Type	cas_meta:Relationship

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ Relationship
☆ lists	*enumeratedEndpoint*	☆ Relationship
☆ Relationship	*enumeratedTargetLink*	☆ targetLink

☆ Relationship	*enumeratedSourceLink*	☆ sourceLink
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ Relationship

8.4.42. ○ revision

A unique string identifying the revision of the entity or relationship instance.

Properties

Element Type	uml:Property

Statements

☆ anElement	*has part*	○ revision
▣ CASCaRA Metamodel Diagram	*shows*	○ revision

8.4.43. ☆ Root

A root class serves to anchor organizer structures in a data package. While a graph has no start and no end, a root instance with a subordinated organizer structure offers a way to explore the graph for a given purpose.
Example:

- A document outline anchored at a root element would define a document to read by auditors and perhaps another one for potential customers.
- When querying all instances of class Root, a list of available documents is returned.

Properties

Element Type	cas_meta:Entity

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ Root
☆ Root	*specializes*	☆ Organizer
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ Root

8.4.44. ☆ shows

A Link class for use by View to enumerate eligible Entity and/or Relationship classes that are visible on a view (e.g. model-diagram).

Properties

Element Type	cas_meta:Link

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ shows
☆ shows	*enumeratedEndpoint*	☆ Model-Element
☆ View	*enumeratedTargetLink*	☆ shows
☆ shows	*specializes*	☆ Link
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ shows

8.4.45. ☆ sourceLink

A Link class for use by View to enumerate eligible Entity and/or Relationship classes that can serve as a source of a Relationship.
By further specializing this class, a speaking title can be given to a sourceLink, such as 'employee' in a 'worksFor' relationship being a specialization of Relationship.

Properties

Element Type	cas_meta:Link	

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ sourceLink
☆ Relationship	*enumeratedSourceLink*	☆ sourceLink
☆ sourceLink	*specializes*	☆ Link
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ sourceLink

8.4.46. ☆ State

A fundamental model-element class for states (e.g. system or process states, information, form, color, ...).

Properties

Element Type	cas_meta:Entity

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ State
☆ State	*specializes*	☆ Model-Element
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ State

8.4.47. ☆ Table

A specialized Organizer with elements (cells) arranged two-dimensionally in columns and rows. Any element SHALL contain a reference to any Entity or Relationship.

Properties

Element Type	cas_meta:Entity

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ Table
☆ Table	*specializes*	☆ Organizer
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ Table

8.4.48. ☆ targetLink

A Link class for use by View to enumerate eligible Entity and/or Relationship classes that can serve as a target of a Relationship.
By further specializing this class, a speaking title can be given to a targetLink, such as 'employer' in a 'worksFor' relationship being a specialization of Relationship.

Properties

Element Type	cas_meta:Link

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ targetLink
☆ Relationship	*enumeratedTargetLink*	☆ targetLink
☆ targetLink	*specializes*	☆ Link
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ targetLink

8.4.49. ○ title

Properties

Element Type	uml:Property

Statements

☆ Identifiable	*has part*	○ title
▣ CASCaRA Metamodel Diagram	*shows*	○ title

8.4.50. ☆ Tree

A specialized Organizer with elements (nodes) arranged in a strictly hierarchical manner. Any element SHALL contain a reference to any Entity or Relationship.

Properties

Element Type	cas_meta:Entity

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ Tree
☆ Outline	*specializes*	☆ Tree
☆ Tree	*specializes*	☆ Organizer
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ Tree

8.4.51. ○ unit

A unit as defined by the International System of Units (SI). A unit can only be defined for a numeric datatype. By default the property has no unit.

Properties

Element Type	uml:Property

Statements

☆ Property	*has part*	○ unit
▣ CASCaRA Metamodel Diagram	*shows*	○ unit

8.4.52. ○ value

The value of a property.
Details:

- If the datatype of the property's class is string, the value is a language string consisting of value and IETF language tag.
- Otherwise the value is a simple value.

Properties

Element Type	uml:Property

Statements

☆ aProperty	*has part*	○ value
▣ CASCaRA Metamodel Diagram	*shows*	○ value

8.4.53. ☆ View

A specialized Organizer with elements displayed in two or three dimensions such as a model-diagram, 3D visualization or pie-chart. Any element SHOULD reference any Entity or Relationship.

Properties

Element Type	cas_meta:Entity

Statements

CASCaRA Semantic Infrastructure	*contains*	☆ View
☆ View	*enumeratedTargetLink*	☆ shows ☆ depicts
☆ View	*specializes*	☆ Organizer
▣ CASCaRA Semantic Infrastructure Diagram	*shows*	☆ View

8.5. CASCaRA Layered Ontology

8.5.1. Introduction

This chapter describes the data entities used in a CASCaRA package or collaboration scope. It covers individual information entities that are relevant as input or output to one or several collaboration use-cases. The entities of the CASCaRA focus on what entities will be required in order to build a digital thread across domains, organizations and IT systems. Therefore, the ontology does not represent entire domain models (e. g. requirements models or CAD models), but relevant entities that can be found inside these models.

8.5.2. Reused Types

Lists all types from other schemas or ontologies used in the ontology.

8.5.2.1. Dublin Core

The Dublin Core schema is a small set of vocabulary terms that can be used to describe digital resources (video, images, web pages, etc.), as well as physical resources such as books or CDs, and objects like artworks. The full set of Dublin Core metadata terms can be found on the Dublin Core Metadata Initiative (DCMI) website. The original set of 15 classic metadata terms, known as the Dublin Core Metadata Element Set (DCMES), is endorsed in a set of standards documents,

8.5.2.1.1. dc:identifier

An unambiguous reference to the resource within a given context.
Recommended practice is to identify the resource by means of a string conforming to an identification system. Examples include International Standard Book Number (ISBN), Digital Object Identifier (DOI), and Uniform Resource Name (URN). Persistent identifiers should be provided as HTTP URIs.

URI	http://purl.org/dc/elements/1.1/identifier

8.5.2.1.2. dc:title

A name given to the resource.

URI	http://purl.org/dc/elements/1.1/title

8.5.2.1.3. dc:description

An account of the resource.
Description may include but is not limited to: an abstract, a table of contents, a graphical representation, or a free-text account of the resource.

URI	http://purl.org/dc/elements/1.1/description

8.5.2.2. XML Schema

8.5.2.2.1. xs:string

URI	http://www.w3.org/2001/XMLSchema#string

8.5.2.2.2. xs:dateTime

URI	http://www.w3.org/2001/XMLSchema#dateTime

8.5.2.2.3. xs:double

URI	http://www.w3.org/2001/XMLSchema#double

8.5.2.2.4. xs:anyURI

URI	http://www.w3.org/2001/XMLSchema#anyURI

8.5.3. Dimensions

Lists all dimensions used in the ontology.

8.5.3.1. Angle

The abstract notion of angle. Narrow concepts include plane angle and solid angle. While both plane angle and solid angle are dimensionless, they are actually length/length and area/area respectively.
Applicable units are those of quantitykind:Angle

URI	http://qudt.org/vocab/quantitykind/Angle
Units	http://qudt.org/vocab/unit/DEG

8.5.3.2. Area

Area is a quantity expressing the two-dimensional size of a defined part of a surface, typically a region bounded by a closed curve.
Applicable units are those of quantitykind:Area

URI	http://qudt.org/vocab/quantitykind/Area
Units	http://qudt.org/vocab/unit/MilliM2

8.5.3.3. Byte Data Volume

particular quantity of data based on a string consisting of 8 bits
Applicable units are those of quantitykind:Count

URI	http://qudt.org/vocab/quantitykind/ByteDataVolume
Units	http://qudt.org/vocab/unit/BYTE http://qudt.org/vocab/unit/KiloBYTE

8.5.3.4. Density

The mass density or density of a material is defined as its mass per unit volume. Mathematically, density is defined as mass divided by volume.
In some cases, density is also defined as its weight per unit volume, although this quantity is more properly called specific weight.
Applicable units are those of quantitykind:Density

URI	http://qudt.org/vocab/quantitykind/Density
Units	http://qudt.org/vocab/unit/KiloGM-PER-M3

8.5.3.5. Electric Current

The quantity kind
is the flow (movement) of electric charge. The amount of electric current through some surface, for example, a section through a copper conductor, is defined as the amount of electric charge flowing through that surface over time. Current is a scalar-valued quantity. Electric current is one of the base quantities in the International System of Quantities, ISQ, on which the International System of Units, SI, is based.
Applicable units are those of quantitykind:ElectricCurrent

URI	http://qudt.org/vocab/quantitykind/ElectricCurrent
Units	http://qudt.org/vocab/unit/A

8.5.3.6. Energy

Energy is the quantity characterizing the ability of a system to do work.
Applicable units are those of quantitykind:Energy

URI	http://qudt.org/vocab/quantitykind/Energy
Units	http://qudt.org/vocab/unit/KiloJ

8.5.3.7. Expansion Ratio

Applicable units are those of quantitykind:ExpansionRatio

URI	http://qudt.org/vocab/quantitykind/ExpansionRatio

Units	http://qudt.org/vocab/unit/PERCENT

8.5.3.8. Force

"Force" is an influence that causes mass to accelerate. It may be experienced as a lift, a push, or a pull. Net force is mathematically equal to the time rate of change of the momentum of the body on which it acts. Since momentum is a vector quantity (has both a magnitude and direction), force also is a vector quantity.
Applicable units are those of quantitykind:Force

URI	http://qudt.org/vocab/quantitykind/Force
Units	http://qudt.org/vocab/unit/N

8.5.3.9. Frequency

"Frequency" is the number of occurrences of a repeating event per unit time. The repetition of the events may be periodic (that is. the length of time between event repetitions is fixed) or aperiodic (i.e. the length of time between event repetitions varies). Therefore, we distinguish between periodic and aperiodic frequencies. In the SI system, periodic frequency is measured in hertz (Hz) or multiples of hertz, while aperiodic frequency is measured in becquerel (Bq). In spectroscopy,
is mostly used. Light passing through different media keeps its frequency, but not its wavelength or wavenumber.
Applicable units are those of quantitykind:Frequency

URI	http://qudt.org/vocab/quantitykind/Frequency
Units	http://qudt.org/vocab/unit/PER-MIN

8.5.3.10. Length

In geometric measurements, length most commonly refers to the longest dimension of an object. In some contexts, the term "length" is reserved for a certain dimension of an object along which the length is measured.
Applicable units are those of quantitykind:Length

URI	http://qudt.org/vocab/quantitykind/Length
Units	http://qudt.org/vocab/unit/MilliM http://qudt.org/vocab/unit/M

8.5.3.11. Linear Thermal Expansion Coefficient

When the temperature of a substance changes, the energy that is stored in the intermolecular bonds between atoms changes. When the stored energy increases, so does the length of the molecular bonds. As a result, solids typically expand in response to heating and contract on cooling; this dimensional response to temperature change is expressed by its coefficient of thermal expansion. Different coefficients of thermal expansion can be defined for a substance depending on whether the expansion is measured by: linear thermal expansion, area thermal expansion, or volumetric thermal expansion.

URI	http://qudt.org/vocab/quantitykind/LinearThermalExpansion
Units	http://qudt.org/vocab/unit/PER-K http://qudt.org/vocab/unit/MilliM-PER-K

8.5.3.12. Mass

In physics, mass, more specifically inertial mass, can be defined as a quantitative measure of an object's resistance to acceleration. The SI unit of mass is the kilogram.
Applicable units are those of quantitykind:Mass

URI	http://qudt.org/vocab/quantitykind/Mass
Units	http://qudt.org/vocab/unit/KiloGM

8.5.3.13. Moment of inertia

The rotational inertia or resistance to change in direction or speed of rotation about a defined axis.

URI	http://qudt.org/vocab/quantitykind/MomentOfInertia
Units	http://qudt.org/vocab/unit/KiloGM-M2

8.5.3.14. Power

Power is the rate at which work is performed or energy is transmitted, or the amount of energy required or expended for a given unit of time.

Applicable units are those of quantitykind:Power

URI	http://qudt.org/vocab/quantitykind/Power
Units	http://qudt.org/vocab/unit/KiloW

8.5.3.15. Rotational Frequency

A measure of the number of cycles that an item revolves per time period.

Applicable units are those of quantitykind:RotationalFrequency

URI	http://qudt.org/vocab/quantitykind/RotationalFrequency
Units	http://qudt.org/vocab/unit/PER-MIN

8.5.3.16. Strain

In any branch of science dealing with materials and their behaviour, strain is the geometrical expression of deformation caused by the action of stress on a physical body. Strain is calculated by first assuming a change between two body states: the beginning state and the final state. Then the difference in placement of two points in this body in those two states expresses the numerical value of strain. Strain therefore expresses itself as a change in size and/or shape.

URI	http://qudt.org/vocab/quantitykind/Strain
Units	http://qudt.org/vocab/unit/PERCENT http://qudt.org/vocab/unit/N-PER-MilliM2

8.5.3.17. Stress

Stress is a measure of the average amount of force exerted per unit area of a surface within a deformable body on which internal forces act. In other words, it is a measure of the intensity or internal distribution of the total internal forces acting within a deformable body across imaginary surfaces. These internal forces are produced between the particles in the body as a reaction to external forces applied on the body.

Applicable units are those of quantitykind:ForcePerArea

URI	http://qudt.org/vocab/quantitykind/Stress
Units	http://qudt.org/vocab/unit/N-PER-MilliM2

8.5.3.18. Torque

In physics, a torque is a vector that measures the tendency of a force to rotate an object about some axis. The magnitude of a torque is defined as force times its lever arm. Just as a force is a push or a pull, a torque can be thought of as a twist. The SI unit for torque is newton meters. In U.S. customary units, it is measured in foot pounds (ft lbf) (also known as "pounds feet"). Mathematically, the torque on a particle (which has the position r in some reference frame) can be defined as the cross product: where, r is the particle's position vector relative to the fulcrum F is the force acting on the particles, or, more generally, torque can be defined as the rate of change of angular momentum: where, L is the angular momentum vector t stands for time.

Applicable units are those of quantitykind:Torque

URI	http://qudt.org/vocab/quantitykind/Torque
Units	http://qudt.org/vocab/unit/N-M

8.5.3.19. Voltage

Voltage, also referred to as Electric Tension, is the difference between electrical potentials of two points. Applicable units are those of quantitykind:Voltage

URI	http://qudt.org/vocab/quantitykind/Voltage
Units	http://qudt.org/vocab/unit/V

8.5.3.20. Volume

The volume of a solid object is the three-dimensional concept of how much space it occupies, often quantified numerically. One-dimensional figures (such as lines) and two-dimensional shapes (such as squares) are assigned zero volume in the three-dimensional space.

Applicable units are those of quantitykind:Volume

URI	http://qudt.org/vocab/quantitykind/Volume
Units	http://qudt.org/vocab/unit/CentiM3

8.5.4. Data Entities

Lists all classes, attributes and relations in the ontology.

8.5.4.1. Organization

Includes all entities representing a corporate or project organization

Figure 2 Icon

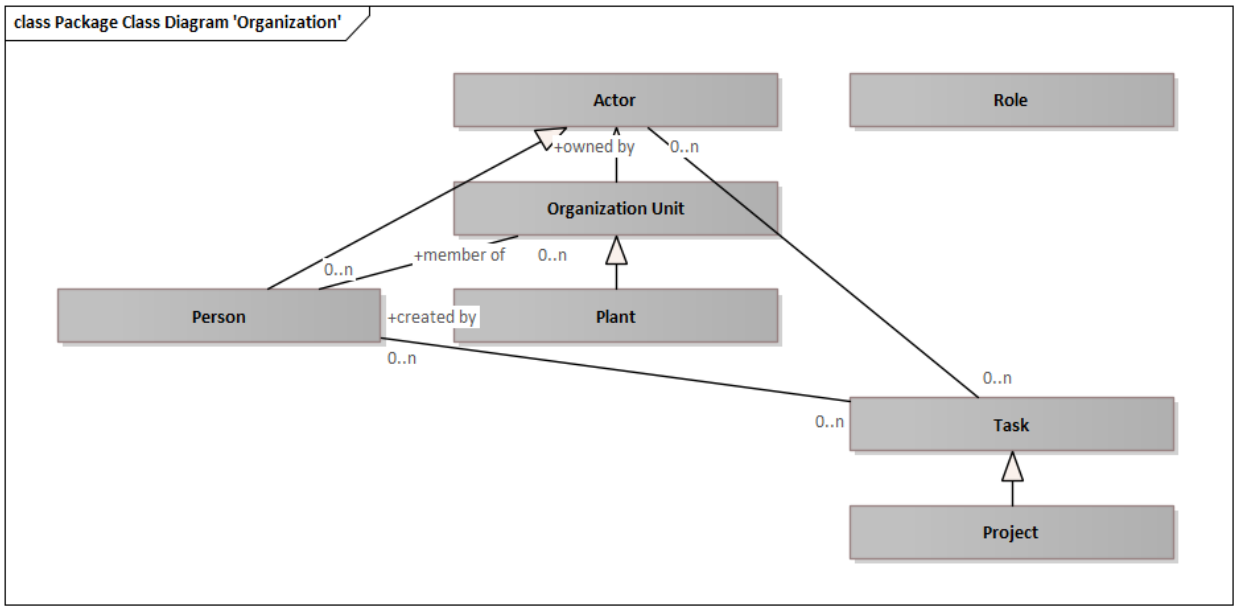


Figure 3 Package Class Diagram 'Organization'

8.5.4.1.1. Role

A defined function to be performed by a project team member, such as testing, filing, inspecting, coding.
https://intranet.prostep.org/display/PROJ/Glossary

Figure 4 Icon

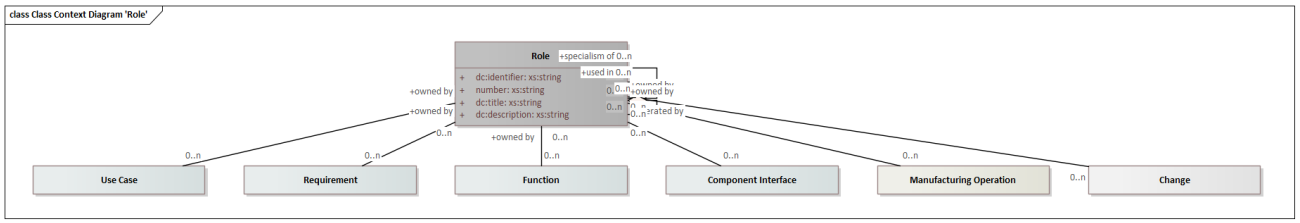


Figure 5 Class Context Diagram 'Role'

Delivered by	Collaborative Product Requirements Management Requirements Engineering Systems Architecture Functional Safety	
Delivered to	Business or Mission Analysis	
Attributes		
Identifier	dc:identifier	Unambiguous identifier of the element
Number		Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Outbound Relations		
part of	Role	Parent element
used in	Role	Context element of the usage
specialism of	Role	More generic item
Inbound Relations		
Role	part of	Parent element
Role	used in	Context element of the usage
Role	specialism of	More generic item
Use Case	owned by	Role owning this element
Requirement	owned by	Role owning this element
Function	owned by	Role owning this element
Component Interface	operated by	Interacting role
Manufacturing Operation	owned by	Role owning this element
Change	owned by	Role owning this element

8.5.4.1.2. Actor

Abstract class for an actor. Derived, specific classes are person and organization unit.

Figure 6 Icon

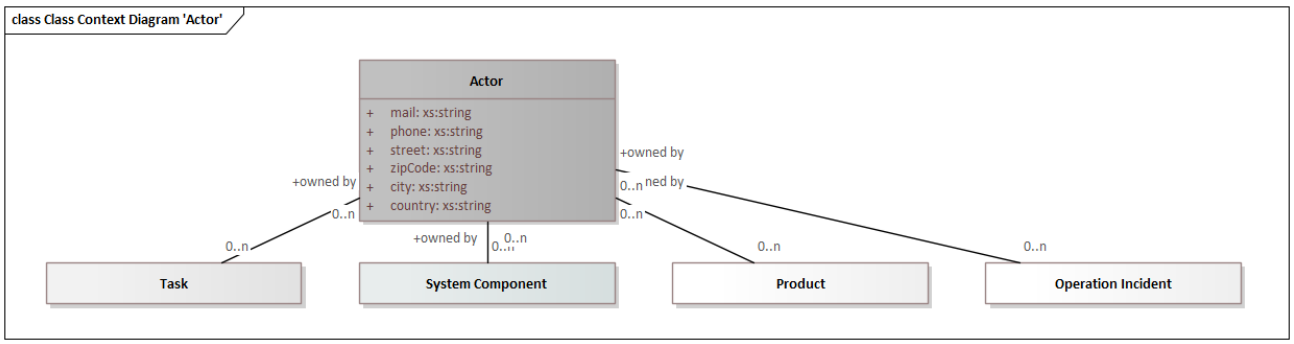


Figure 7 Class Context Diagram 'Actor'

Specializations	Organization Unit Person	
Attributes		
Identifier	dc:identifier	Unambiguous identifier of the element
Mail		E-mail address of the actor
Phone		Phone number of the actor
Street		Street of the actor
ZIP Code		ZIP code number of the actor
City		City of the actor
Country		Country of the actor

8.5.4.1.3. Person

Represents a real person, but may also be a system user that appears in source data. The FOAF (Friend of a Friend) Ontology has been considered, but is not used as it does not provide a structured representation for firstname, lastname and title of a person.

Figure 8 Icon

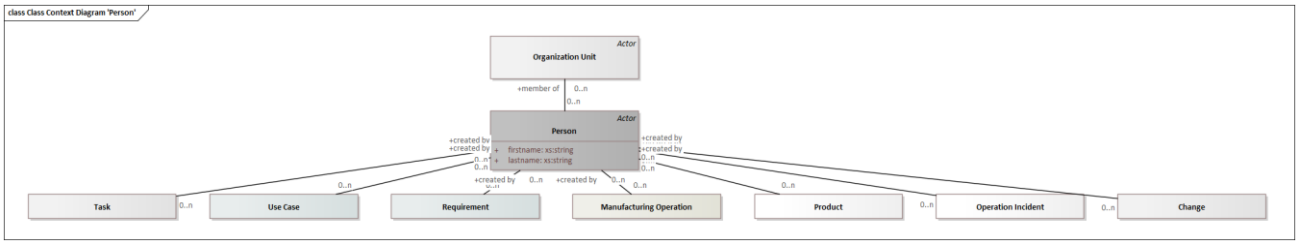


Figure 9 Class Context Diagram 'Person'

Generalizations	Actor	
Attributes		
Number	xs:string	Legible number or name of the element, i. e. part number
Description	dc:description	Description of the element
Title	dc:title	Title or degree of the person
Firstname		First name of the person
Lastname		Last name of the person

Outbound Relations

member of	Organization Unit	Parent organization unit

8.5.4.1.4. Organization Unit

An organization unit is part of a corporate structure. It comprises persons as well as sub-organization units.

Figure 10 Icon

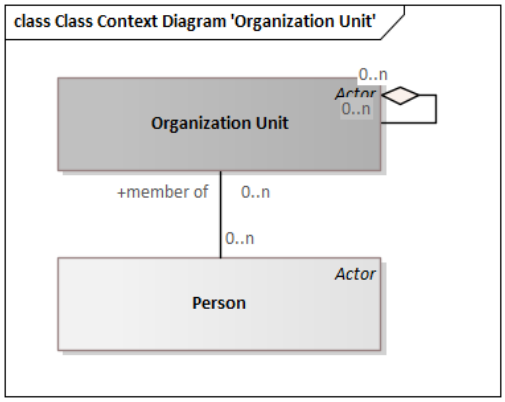


Figure 11 Class Context Diagram 'Organization Unit'

Generalizations	Actor	
Specializations	Plant	

Attributes

Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element

Outbound Relations

part of	Organization Unit	Parent element

Inbound Relations

Organization Unit	part of	Parent element
Person	member of	Parent organization unit

8.5.4.1.5. Plant

A plant that provides manufacturing capabilities and resources and executes manufacuring operations.

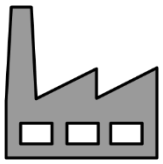


Figure 12 Icon

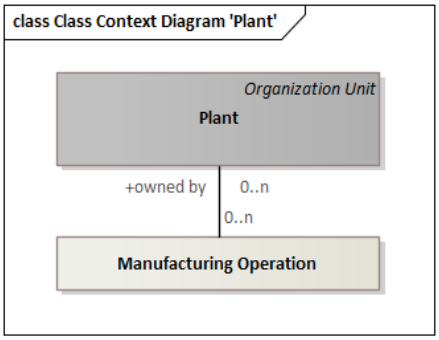


Figure 13 Class Context Diagram 'Plant'

Generalizations	Organization Unit

8.5.4.1.6. Task

A task is a definite piece of work assigned to a person or organisation

Figure 14 Icon

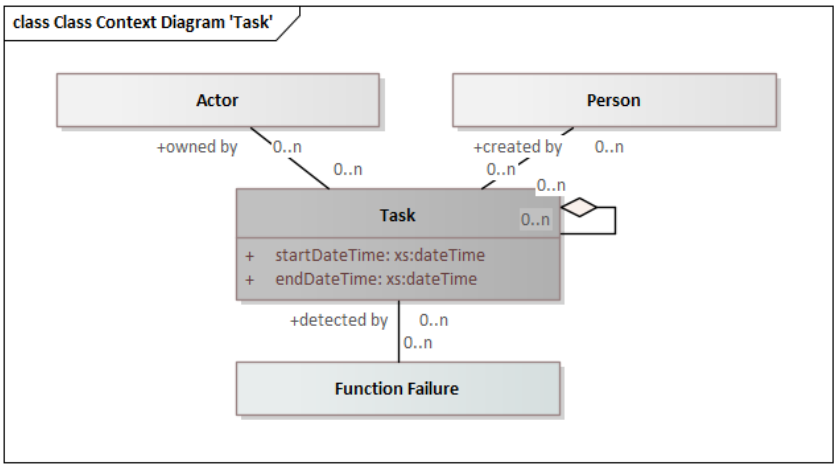


Figure 15 Class Context Diagram 'Task'

Specializations	Project	
Attributes		
Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
StartDateTime		Start date and time of the element
EndDateTime		End date and time of the element
Outbound Relations		
part of	Task	Parent element
created by	Person	The person who created this element

owned by	Actor	Actor owning this element

Inbound Relations

Task	part of	Parent element
Function Failure	detected by	Detection task for the function failure

8.5.4.1.7. Project

Represents a project or sub-project

Figure 16 Icon

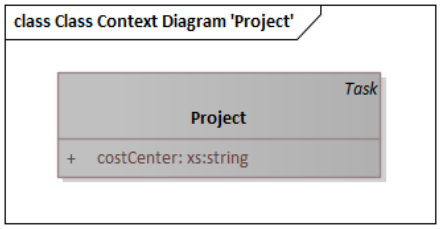


Figure 17 Class Context Diagram 'Project'

Generalizations	Task

Attributes

Cost center	Cost center of the project

8.5.4.2. Product Architecture

Includes all entities contributing to the product architecture

Figure 18 Icon

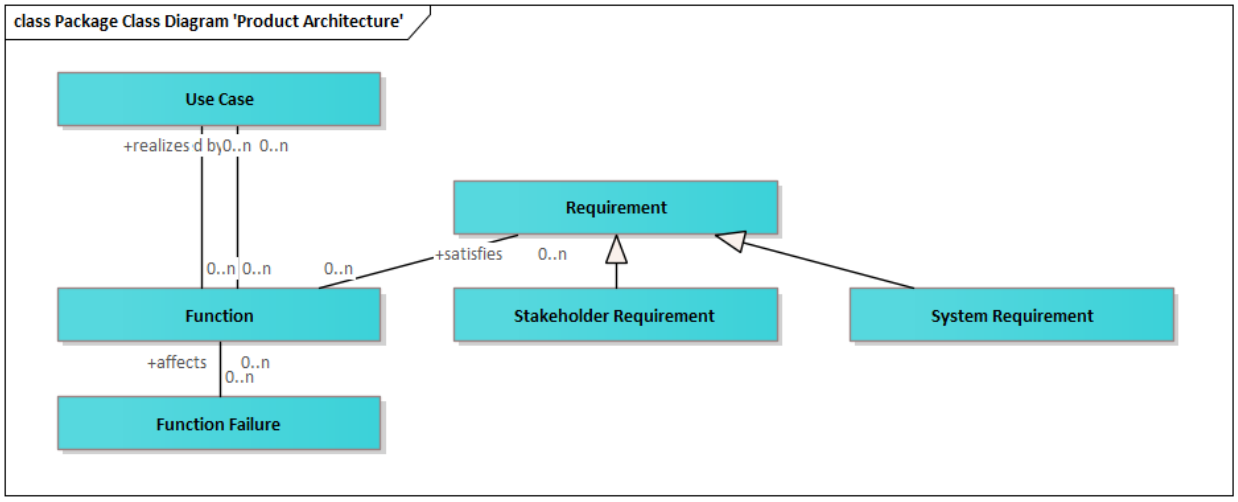


Figure 19 Package Class Diagram 'Product Architecture'

8.5.4.2.1. Use Case

Specifies what the user of a product will do in order to achieve a goal

Figure 20 Icon

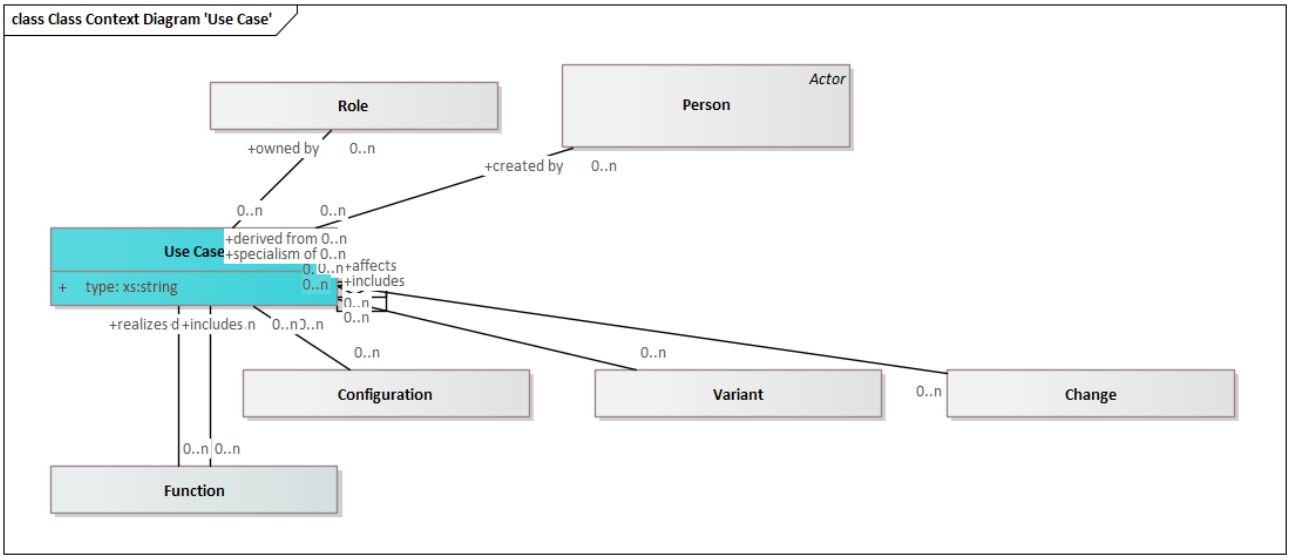


Figure 21 Class Context Diagram 'Use Case'

Related Standards	VDI 2519 Blatt 1 Vorgehensweise bei der Erstellung von Lasten-/Pflichtenheften
Delivered by	Collaborative Product Requirements Management Requirements Engineering Systems Architecture Functional Safety Product Usage Maintenance & Service

Delivered to	Business or Mission Analysis	
Attributes		
Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type		Type of an element
Outbound Relations		
part of	Use Case	Parent element
specialism of	Use Case	More generic item
created by	Person	The person who created this element
owned by	Role	Role owning this element
derived from	Use Case	
Inbound Relations		
Use Case	part of	Parent element
Use Case	specialism of	More generic item
Use Case	derived from	
Function	realizes	Use cases realized with this function
Function	owned by	
Configuration	includes	Entities included in this configuration.
Variant	includes	Entities included in this variant.
Change	affects	Entities affected by this change.

8.5.4.2.2. Requirement

Characteristics that identify the accomplishment levels needed to achieve specific objectives for a given set of conditions. Requirements can be 'must have', 'can have' and 'nice to have'.
A. A condition or capability needed by a user to solve a problem or achieve an objective.
B. A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed document.
C. A documented representation of a condition or capability as in definition (A:) or (B).
https://intranet.prostep.org/display/PROJ/Glossary

Figure 22 Icon

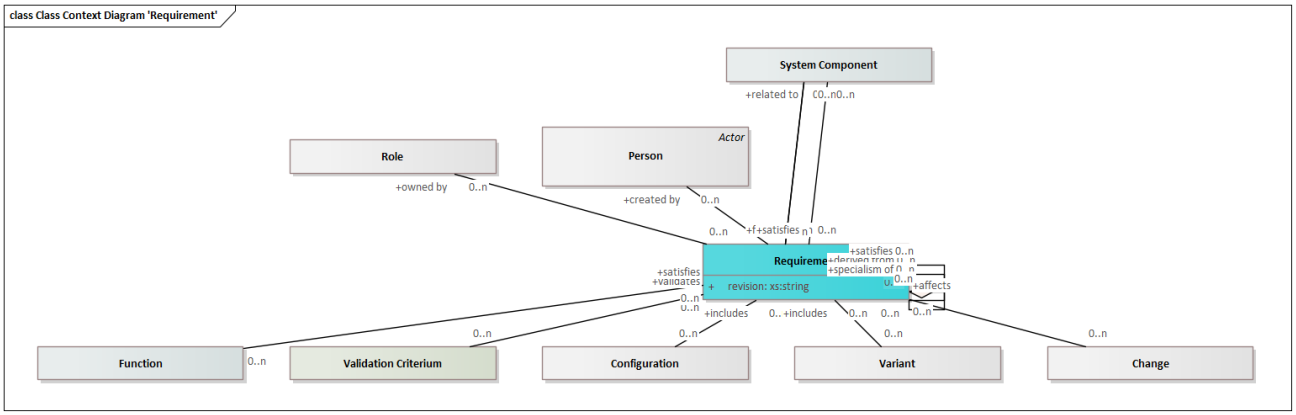


Figure 23 Class Context Diagram 'Requirement'

Specializations	Stakeholder Requirement System Requirement
Related Standards	VDI 2519 Blatt 1 Vorgehensweise bei der Erstellung von Lasten-/Pflichtenheften ISO/IEC/IEEE 29148 Requirements Engineering CSA ISO/IEC/IEEE 15288 System life cycle processes
Delivered to	Collaborative Product Requirements Management Requirements Engineering

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element
Revision		Revisionskennung des Elements

Outbound Relations

part of	Requirement	Parent element
specialism of	Requirement	More generic item
created by	Person	The person who created this element
owned by	Role	Role owning this element
derived from	Requirement	
satisfies	Requirement	Requirement satisfied by this requirement. This is typically used for a stakeholder requirement satisfied by a system requirement
related to	System Component	System Component this requirement relates to.

Inbound Relations

Requirement	part of	Parent element
Requirement	specialism of	More generic item
Requirement	derived from	
Requirement	satisfies	Requirement satisfied by this requirement. This is typically used for a stakeholder requirement satisfied by a system requirement
Function	satisfies	Requirement satisfied by this function
System Component	satisfies	Requirement satisfied by this system component
System Component	fulfils	Requirement implemented by the component. According to systems engineering best-practice, a requirement should first

		be answered by a function, and the function will be implemented by a component. This direct relation should therefore only be used if no functions have been defined.
Validation Criterium	validates	Requirement validated by this validation criterium
Configuration	includes	Entities included in this configuration.
Variant	includes	Entities included in this variant.
Change	affects	Entities affected by this change.

8.5.4.2.3. Stakeholder Requirement

Figure 24 Icon

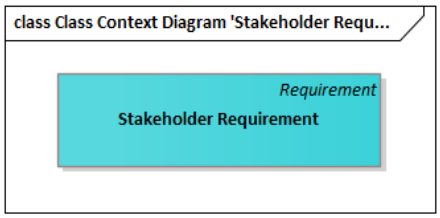


Figure 25 Class Context Diagram 'Stakeholder Requirement'

Generalizations	Requirement
Related Standards	CSA ISO/IEC/IEEE 15288 System life cycle processes
Delivered by	Systems Architecture Long-Term Archiving
Delivered to	Collaborative Product Requirements Management

8.5.4.2.4. System Requirement

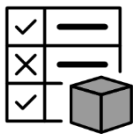


Figure 26 Icon

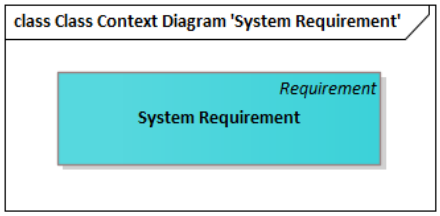


Figure 27 Class Context Diagram 'System Requirement'

Generalizations	Requirement
Related Standards	CSA ISO/IEC/IEEE 15288 System life cycle processes ISO/IEC 20246 Bewertungen von Arbeitsergebnissen

Delivered by	Functional Safety Long-Term Archiving Mechanical Design Define Joining Elements Electrical/Electronical Design Software Engineering Plan and prepare Simulation Prepare Testing Design product line
Delivered to	Collaborative Architecture Design Co-Development

8.5.4.2.5. Function

A system, product or component function describes the way a system works or performs a specific task.

Figure 28 Icon

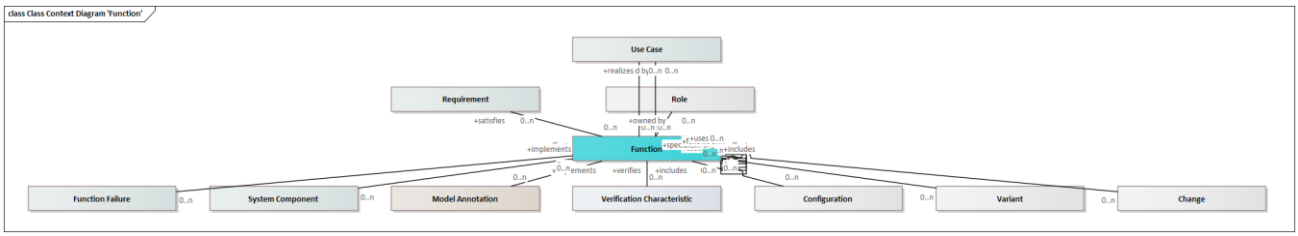


Figure 29 Class Context Diagram 'Function'

Related Standards	VDI 2519 Blatt 1 Vorgehensweise bei der Erstellung von Lasten-/Pflichtenheften CSA ISO/IEC/IEEE 15288 System life cycle processes ISO/IEC 20246 Bewertungen von Arbeitsergebnissen
Delivered by	Functional Safety Product Usage Maintenance & Service Long-Term Archiving Mechanical Design Electrical/Electronical Design Software Engineering Plan and prepare Simulation Prepare Testing Design product line Disassembly & Recycling
Delivered to	Systems Architecture Functional Safety Collaborative Architecture Design Co-Development

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element
Revision	xs:string	Revisionskennung des Elements

Outbound Relations

part of	Function	Parent element
used in	Function	Context element of the usage
specialism of	Function	More generic item
owned by	Role	Role owning this element
satisfies	Requirement	Requirement satisfied by this function
follows	Function	In a sequence of product functions, specifies the previous product function.
uses	Function	Function used by this Function
realizes	Use Case	Use cases realized with this function
owned by	Use Case	

Inbound Relations

Function	part of	Parent element
Function	used in	Context element of the usage
Function	specialism of	More generic item
Function	follows	In a sequence of product functions, specifies the previous product function.
Function	uses	Function used by this Function
System Component	implements	Product function implemented by the component
Function Failure	affects	Product function affected by this product function failure
Model Annotation	implements	Product function implemented by the model annotation
Verification Characteristic	verifies	Function verified by this verification characteristic
Configuration	includes	Entities included in this configuration.
Variant	includes	Entities included in this variant.
Change	affects	Entities affected by this change.

8.5.4.2.6. Function Failure

Represents a system, product or component failure. A failure negates a corresponding function. Malfunctions are typically managed in an FMEA.

Figure 30 Icon

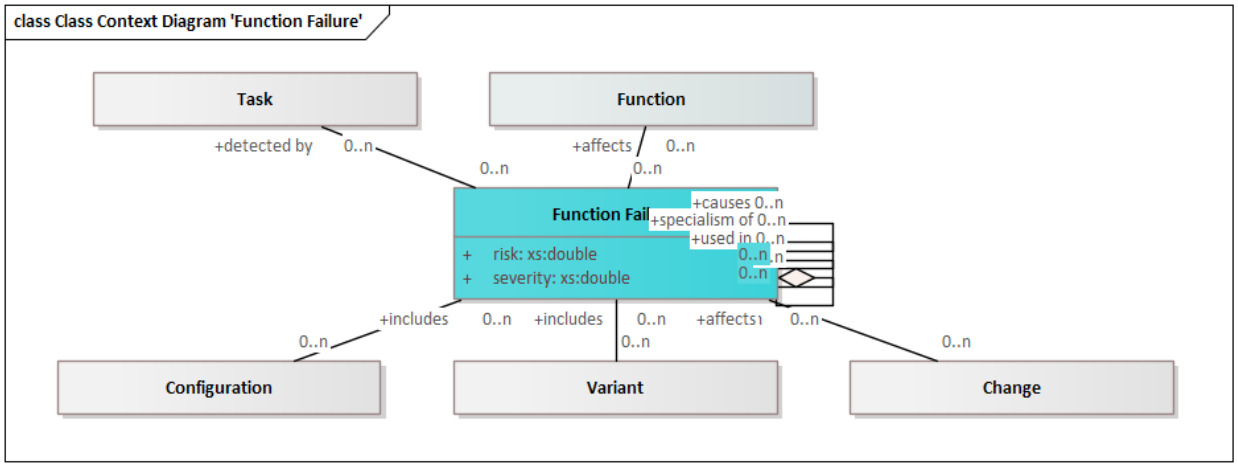


Figure 31 Class Context Diagram 'Function Failure'

Related Standards	ISO/IEC 20246 Bewertungen von Arbeitsergebnissen ISO 26262 Funktionale Sicherheit ISO 21448 Safety of the intended functionality ARP4754B Guidelines for Development of Civil Aircraft and Systems ARP4761A Guidelines for Conducting the Safety Assessment Process on Civil Aircraft, Systems, and Equipment DIN EN 61508-1 Part 1: General requirements
Delivered by	Long-Term Archiving Inspection Planning
Delivered to	Systems Architecture Functional Safety Collaborative Architecture Design Co-Development

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element
Revision	xs:string	Revisionskennung des Elements
Risk		Risk of a function failure to occur.
Severity		Severity of a function failure when it occurs.

Outbound Relations

part of	Function Failure	Parent element
used in	Function Failure	Context element of the usage
specialism of	Function Failure	More generic item
affects	Function	Product function affected by this product function failure
causes	Function Failure	Product function failure caused by this product function failure
detected by	Task	Detection task for the function failure

Inbound Relations

Function Failure	part of	Parent element
Function Failure	used in	Context element of the usage
Function Failure	specialism of	More generic item
Function Failure	causes	Product function failure caused by this product function failure

Configuration	includes	Entities included in this configuration.
Variant	includes	Entities included in this variant.
Change	affects	Entities affected by this change.

8.5.4.3. Product Design

Comprises all entities for domain-specific product design, including mechanical, electric/electronic and software design.

Figure 32 Icon

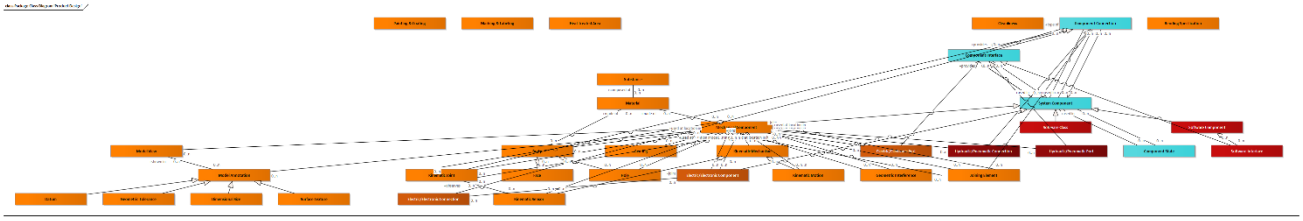


Figure 33 Package Class Diagram 'Product Design'

8.5.4.3.1. Systems Design

Includes all entities relevant to the design of a system. The system level is intended to be generic and will find its specific implementation in mechanical design, electric/electronic design or software design.

Figure 34 Icon

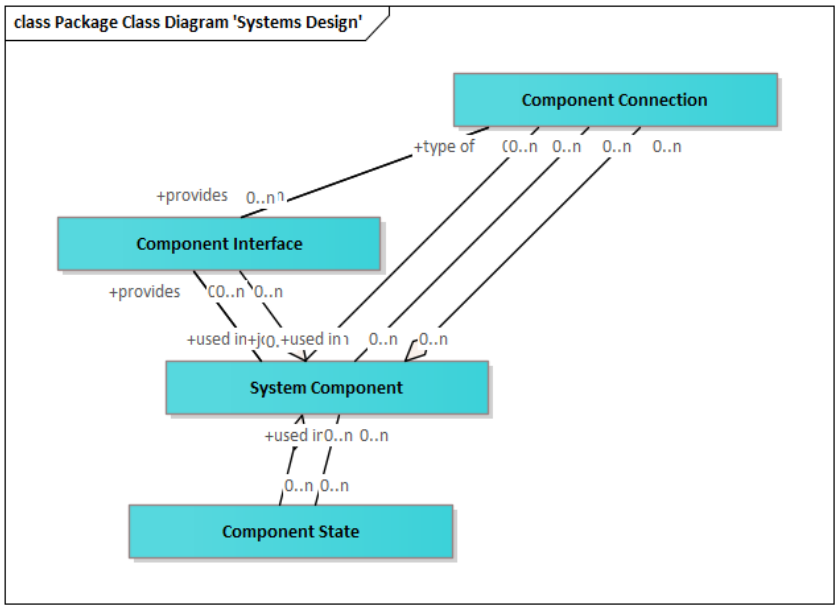


Figure 35 Package Class Diagram 'Systems Design'

8.5.4.3.1.1. System Component

Represents a generic component of a system

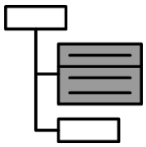


Figure 36 Icon

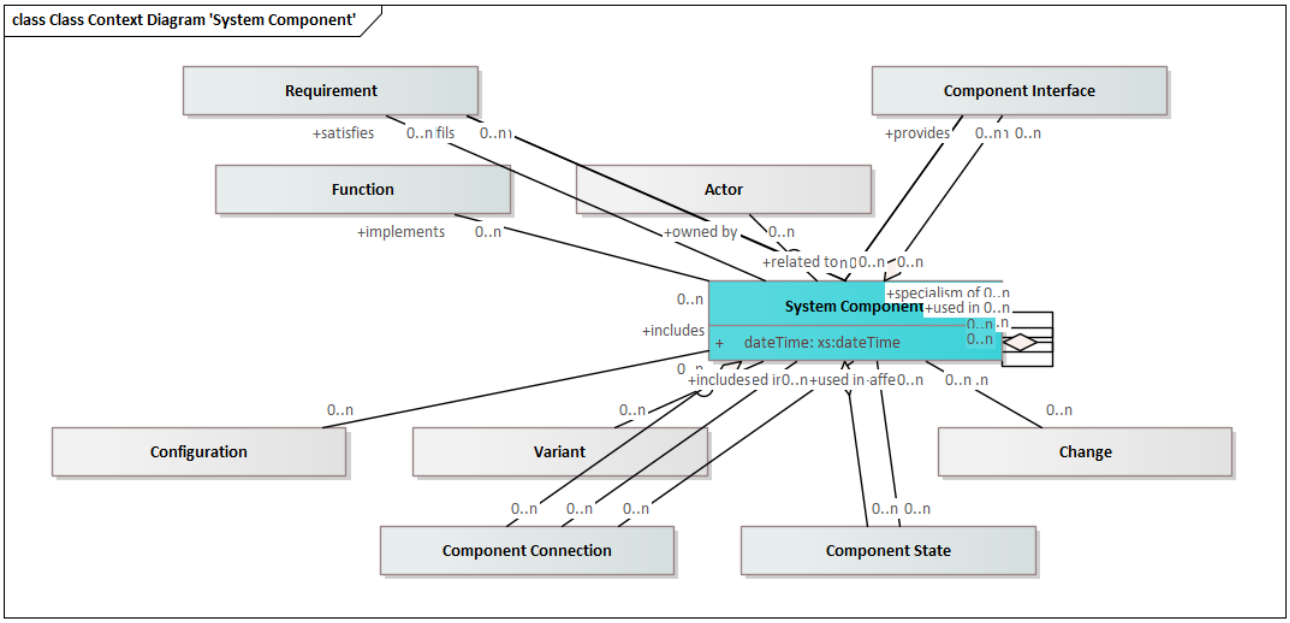


Figure 37 Class Context Diagram 'System Component'

Specializations	Mechanical Component Electric/Electronic Component Software Class Software Component
Related Standards	CSA ISO/IEC/IEEE 15288 System life cycle processes ISO/IEC 20246 Bewertungen von Arbeitsergebnissen ISO/IEC/IEEE 42010 Software, systems and enterprise - Architecture description
Delivered by	Functional Safety Long-Term Archiving Mechanical Design Electrical/Electronical Design Software Engineering Design product line Design product asset instance Definition product asset instance configurability space
Delivered to	Systems Architecture Functional Safety Collaborative Architecture Design Co-Development

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element

Description	dc:description	Description of the element
Type	xs:string	Type of an element
DateTime	xs:dateTime	Date and time of the element
Revision	xs:string	Revisionskennung des Elements
Outbound Relations		
part of	System Component	Parent element
used in	System Component	Context element of the usage
specialism of	System Component	More generic item
owned by	Actor	Actor owning this element
implements	Function	Product function implemented by the component
satisfies	Requirement	Requirement satisfied by this system component
fulfils	Requirement	Requirement implemented by the component. According to systems engineering best-practice, a requirement should first be answered by a function, and the function will be implemented by a component. This direct relation should therefore only be used if no functions have been defined.
provides	Component Interface	
Inbound Relations		
System Component	part of	Parent element
System Component	used in	Context element of the usage
System Component	specialism of	More generic item
Component Interface	part of	Parent system component
Component Interface	used in	Context system component of the usage
Component Connection	part of	Parent system component
Component Connection	used in	Context system component of the usage
Component Connection	joins	Represents the system components joined by this component connection.
Requirement	related to	System Component this requirement relates to.
Component State	part of	Parent system component
Component State	used in	Context system component of the usage
Configuration	includes	Entities included in this configuration.
Variant	includes	Entities included in this variant.
Change	affects	Entities affected by this change.

8.5.4.3.1.2. Component Interface

Represents a generic interface of a system component

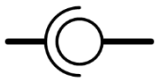


Figure 38 Icon

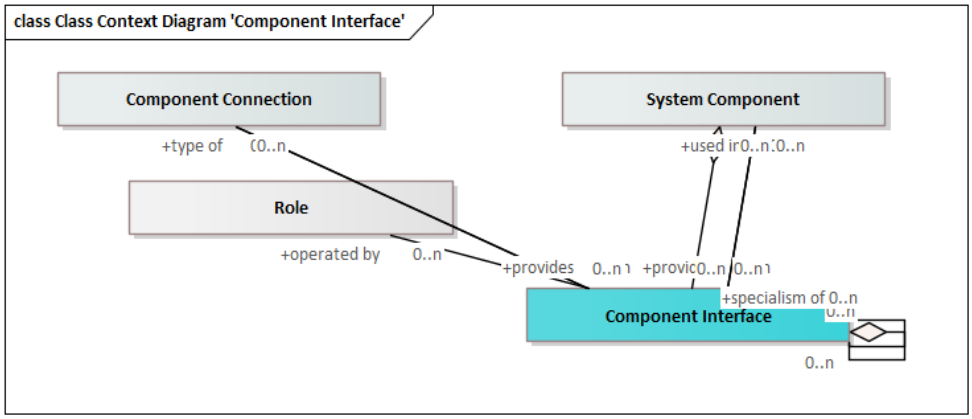


Figure 39 Class Context Diagram 'Component Interface'

Specializations	Electric/Electronic Port Software Interface Hydraulic/Pneumatic Port
Related Standards	ISO/IEC 20246 Bewertungen von Arbeitsergebnissen
Delivered by	Functional Safety Long-Term Archiving Mechanical Design Electrical/Electronical Design Software Engineering Design product line Design product asset instance Definition product asset instance configurability space
Delivered to	Systems Architecture Functional Safety Collaborative Architecture Design Co-Development

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element
DateTime	xs:dateTime	Date and time of the element

Outbound Relations

part of	Component Interface	Parent element
part of	System Component	Parent system component
used in	System Component	Context system component of the usage
specialism of	Component Interface	More generic item
operated by	Role	Interacting role
type of	Component Connection	

Inbound Relations

Component Interface	part of	Parent element
Component Interface	specialism of	More generic item
Component Connection	provides	
System Component	provides	

8.5.4.3.1.3. Component Connection

Represents a connection between two components through interfaces

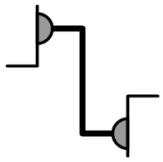


Figure 40 Icon

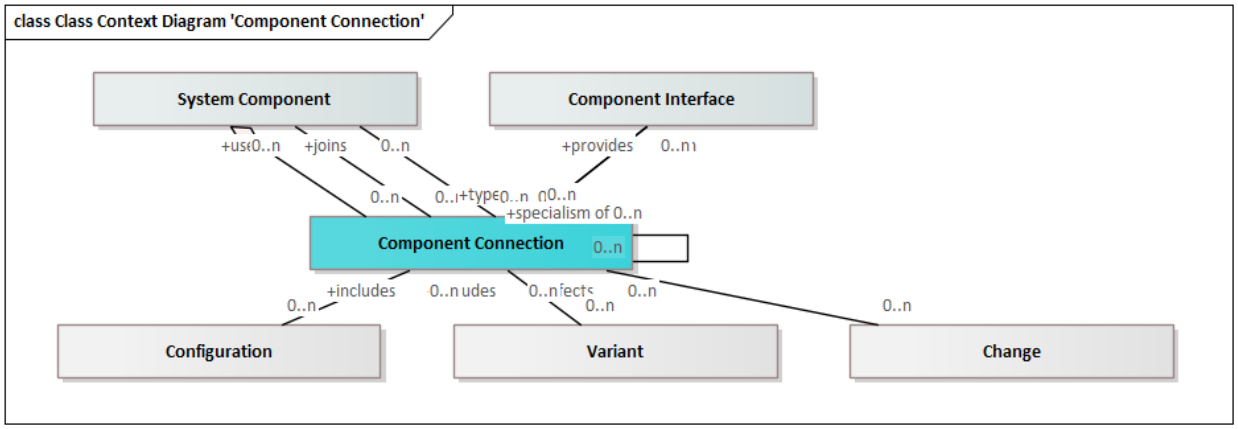


Figure 41 Class Context Diagram 'Component Connection'

Specializations	Kinematic Joint Joining Element Electric/Electronic Connection Hydraulic/Pneumatic Connection
Related Standards	ISO/IEC 20246 Bewertungen von Arbeitsergebnissen
Delivered by	Functional Safety Long-Term Archiving Mechanical Design Electrical/Electronical Design Software Engineering Design product line Design product asset instance Definition product asset instance configurability space
Delivered to	Systems Architecture Functional Safety Collaborative Architecture Design Co-Development

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element
DateTime	xs:dateTime	Date and time of the element

Outbound Relations

part of	System Component	Parent system component
used in	System Component	Context system component of the usage
specialism of	Component Connection	More generic item

provides	Component Interface	
joins	System Component	Represents the system components joined by this component connection.

Inbound Relations

Component Connection	specialism of	More generic item
Component Interface	type of	
Configuration	includes	Entities included in this configuration.
Variant	includes	Entities included in this variant.
Change	affects	Entities affected by this change.

8.5.4.3.1.4. Component State

Represents the condition of a system component. Can be it a value, a condition, an information storage or a physical shape.

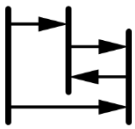


Figure 42 Icon

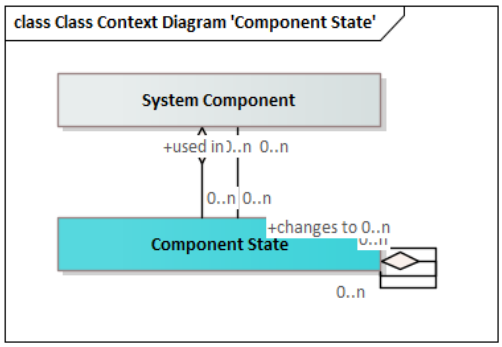


Figure 43 Class Context Diagram 'Component State'

Related Standards	ISO/IEC 20246 Bewertungen von Arbeitsergebnissen
Delivered by	Functional Safety Long-Term Archiving Mechanical Design Electrical/Electronical Design Software Engineering Design product line Design product asset instance Definition product asset instance configurability space
Delivered to	Systems Architecture Functional Safety Collaborative Architecture Design Co-Development

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element
DateTime	xs:dateTime	Date and time of the element

Outbound Relations

part of	Component State	Parent element
part of	System Component	Parent system component
used in	System Component	Context system component of the usage
changes to	Component State	Target state that is reached starting from an initial state

Inbound Relations

Component State	part of	Parent element
Component State	changes to	Target state that is reached starting from an initial state

8.5.4.3.2. Mechanical Design

Includes all entities relevant to mechanical design

Figure 44 Icon

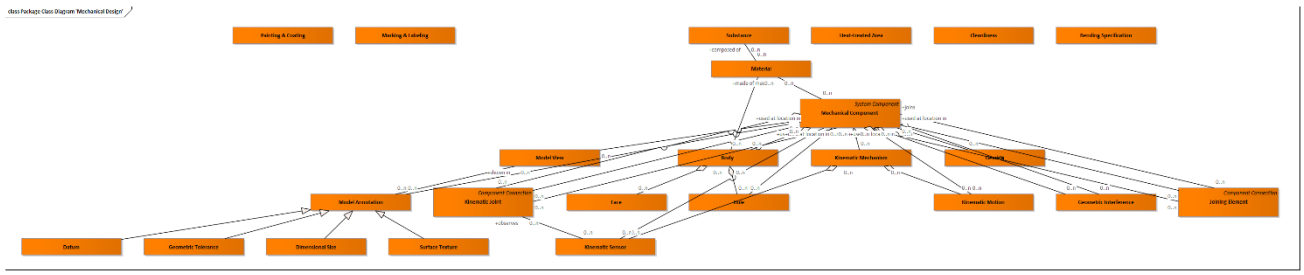


Figure 45 Package Class Diagram 'Mechanical Design'

8.5.4.3.2.1. Mechanical Component

Represents a single mechanical part, an assembly or the entire product

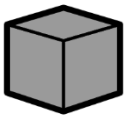


Figure 46 Icon

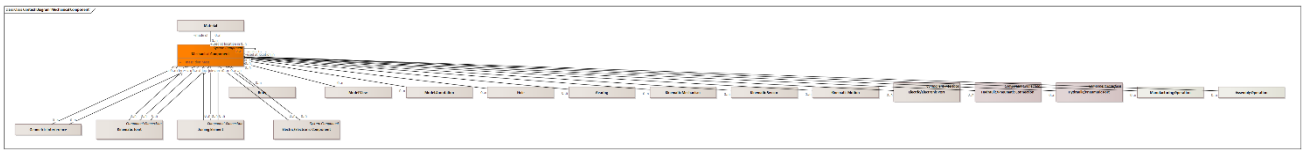


Figure 47 Class Context Diagram 'Mechanical Component'

Generalizations	System Component
Specializations	Manufacturing Equipment Component Verification Resource
Delivered by	Product Usage Maintenance & Service Long-Term Archiving Define Joining Elements Electrical/Electronical Design Software Engineering

	Plan and prepare Simulation Prepare Testing Disassembly & Recycling Inspection Planning Electrical/Mechanical Design Collaboration Perform and document Simulation Perform and document Testing Process Planning Cost Calculation Plant Layout Tools & Equipment Design Parts Manufacturing Product Assembly Quality Inspection Packaging & Logistics Procurement	
Delivered to	Mechanical Design	
Attributes		
Mass		Mass of the mechanical component
Outbound Relations		
used at location in	Mechanical Component	Context element of a located usage
made of	Material	Material assigned to the mechanical component
Inbound Relations		
Mechanical Component	used at location in	Context element of a located usage
Body	part of	Parent mechanical component
Geometric Interference	part of	Parent mechanical component
Geometric Interference	interferes with	Mechanical component instance this interference relates to
Model View	part of	Parent mechanical component
Model Annotation	part of	Parent mechanical component
Hole	used at location in	Context element of a located usage
Gearing	used at location in	Context element of a located usage
Kinematic Mechanism	part of	Parent mechanical component
Kinematic Joint	part of	Parent mechanical component
Kinematic Joint	used at location in	Context element of a located usage
Kinematic Joint	joins	Represents the system components joined by this component connection.
Kinematic Sensor	used at location in	Context element of a located usage
Kinematic Motion	involves	Components involved by the kinematic motion
Joining Element	part of	Parent mechanical component
Joining Element	used at location in	Context element of a located usage
Joining Element	joins	Represents the system components joined by this component connection.
Manufacturing Operation	applies to	Mechanical or electrical component this manufacturing operation applies to
Electric/Electronic Component	used at location in	Context element of a located usage
Electric/Electronic Component	relates to	Mechanical component this electric/electronic component relates to
Electric/Electronic Port	used at location in	Context element of a located usage

Hydraulic/Pneumatic Connection	joins	Represents the system components joined by this component connection.
Hydraulic/Pneumatic Port	used at location in	Context element of a located usage
Assembly Operation	adds	Mechanical or electrical components added by this manufacturing operation

8.5.4.3.2.2. Body

Geometric representation of the component

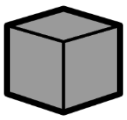


Figure 48 Icon

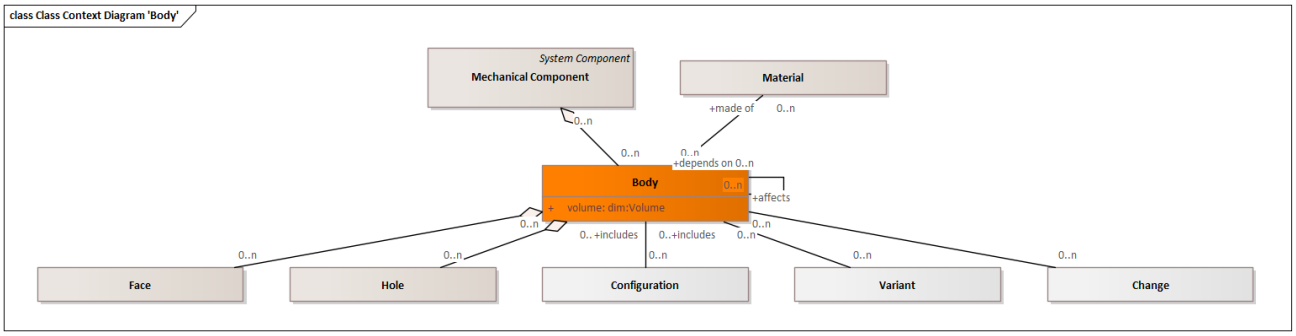


Figure 49 Class Context Diagram 'Body'

Delivered by	Maintenance & Service Long-Term Archiving Electrical/Electronical Design Software Engineering Plan and prepare Simulation Prepare Testing Disassembly & Recycling Inspection Planning Electrical/Mechanical Design Collaboration Perform and document Simulation Process Planning Cost Calculation Plant Layout Tools & Equipment Design Parts Manufacturing Product Assembly Quality Inspection Packaging & Logistics Procurement
Delivered to	Mechanical Design

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Title	dc:title	Title of the element
Type	xs:string	Type of an element
Volume		Geometric volumen of the element

Outbound Relations

part of	Mechanical Component	Parent mechanical component
depends on	Body	Body this body depends on
made of	Material	Material assigned to the mechanical component
Inbound Relations		
Body	depends on	Body this body depends on
Face	part of	Parent body
Hole	part of	Parent body
Configuration	includes	Entities included in this configuration.
Variant	includes	Entities included in this variant.
Change	affects	Entities affected by this change.

8.5.4.3.2.3. Face

Face on the geometric representation of the component

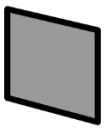


Figure 50 Icon

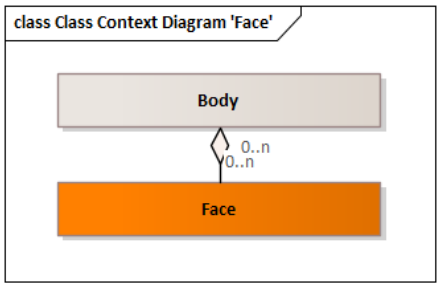


Figure 51 Class Context Diagram 'Face'

Related Standards	ISO 7499 Unique integral feature identification (UIFI)
Delivered by	Define Joining Elements

Attributes		
Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Type	xs:string	Type of an element
Outbound Relations		
part of	Body	Parent body

8.5.4.3.2.4. Geometric Interference

Interference (proximity, contact or clash) between two component instances

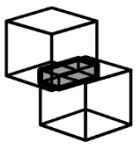


Figure 52 Icon

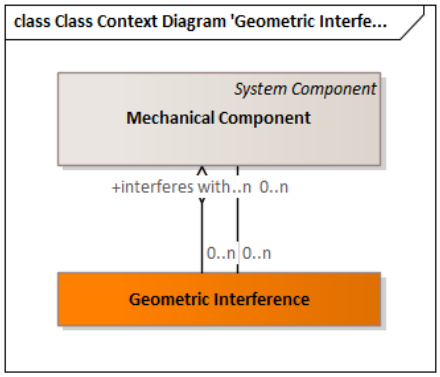


Figure 53 Class Context Diagram 'Geometric Interference'

Delivered by	Plan and prepare Simulation	
Delivered to	Mechanical Design	
Attributes		
Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element
Outbound Relations		
part of	Mechanical Component	Parent mechanical component
interferes with	Mechanical Component	Mechanical component instance this interference relates to

8.5.4.3.2.5. Material

Material assigned to the mechanical component

Figure 54 Icon

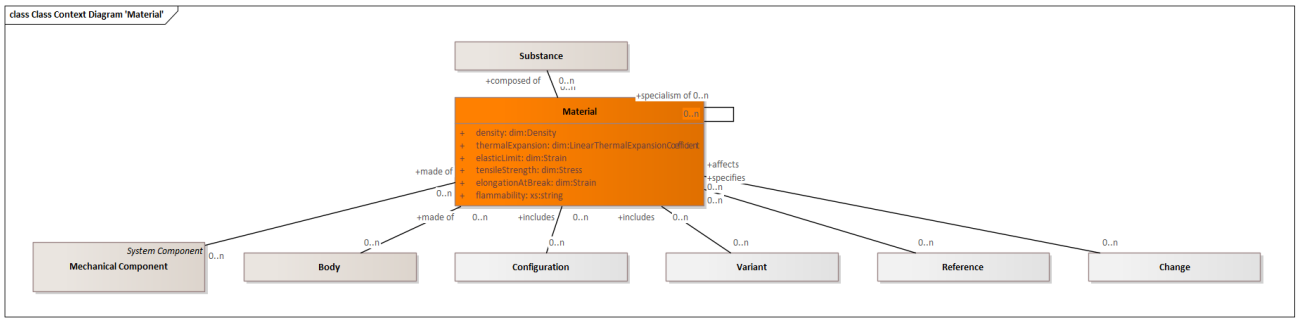


Figure 55 Class Context Diagram 'Material'

Related Standards	VDA 231-200 Werkstoffdatensatz - Spezifikation von Werkstoffen und Oberflächen in IT-Systemen VDA 231-106 Werkstoff-Klassifizierung im Kraftfahrzeugbau: Aufbau und Nomenklatur
Delivered by	Product Usage Maintenance & Service Long-Term Archiving

	Define Joining Elements Plan and prepare Simulation Prepare Testing Disassembly & Recycling Inspection Planning Perform and document Simulation Perform and document Testing Process Planning Cost Calculation Tools & Equipment Design Parts Manufacturing Product Assembly Packaging & Logistics Procurement	
Delivered to	Mechanical Design	
Attributes		
Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element
Density		Specific density of the material
Thermal Expansion		Thermal expansion of the material
Elastic Limit		Elastic limit of the material
Tensile Strength		Tensile strength of the material
Elongation at Break		Elongation of the material at break
Flammability		Flammability of the material
Outbound Relations		
specialism of	Material	More generic item
composed of	Substance	Substances this material is composed of
Inbound Relations		
Material	specialism of	More generic item
Mechanical Component	made of	Material assigned to the mechanical component
Body	made of	Material assigned to the mechanical component
Configuration	includes	Entities included in this configuration.
Variant	includes	Entities included in this variant.
Reference	specifies	Entities specified by this reference.
Change	affects	Entities affected by this change.

8.5.4.3.2.6. Substance

Chemical substance a material is composed of

Figure 56 Icon

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element

Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Type	xs:string	Type of an element

8.5.4.3.2.7. Model View

Represents a view of product geometry, including the definition of the viewpoint, and the visibility of components and annotations

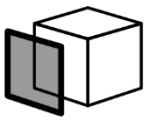


Figure 57 Icon

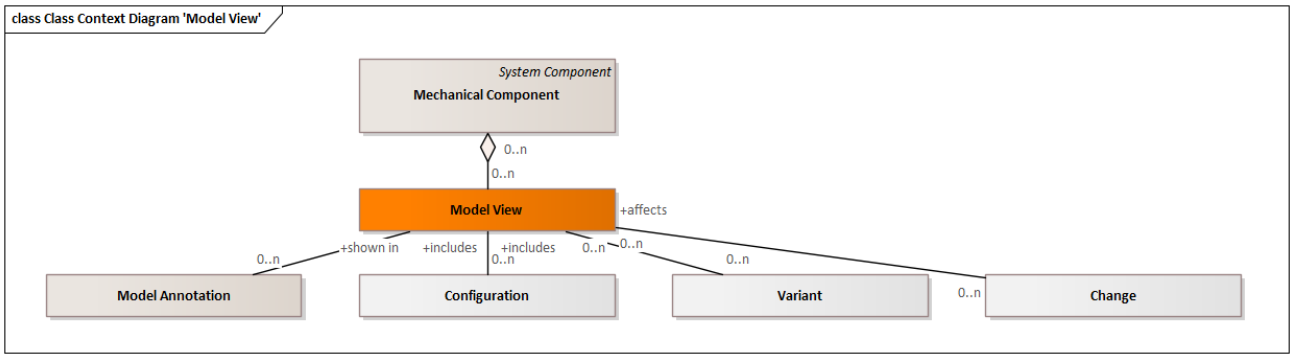


Figure 58 Class Context Diagram 'Model View'

Delivered by	Inspection Planning Process Planning Cost Calculation Plant Layout Tools & Equipment Design Parts Manufacturing Product Assembly	
Delivered to	Mechanical Design	
Attributes		
Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element
Outbound Relations		
part of	Mechanical Component	Parent mechanical component

8.5.4.3.2.8. Model Annotation

A text annotation, dimension, tolerance, surface specification, etc. with relation to product geometry

Figure 59 Icon

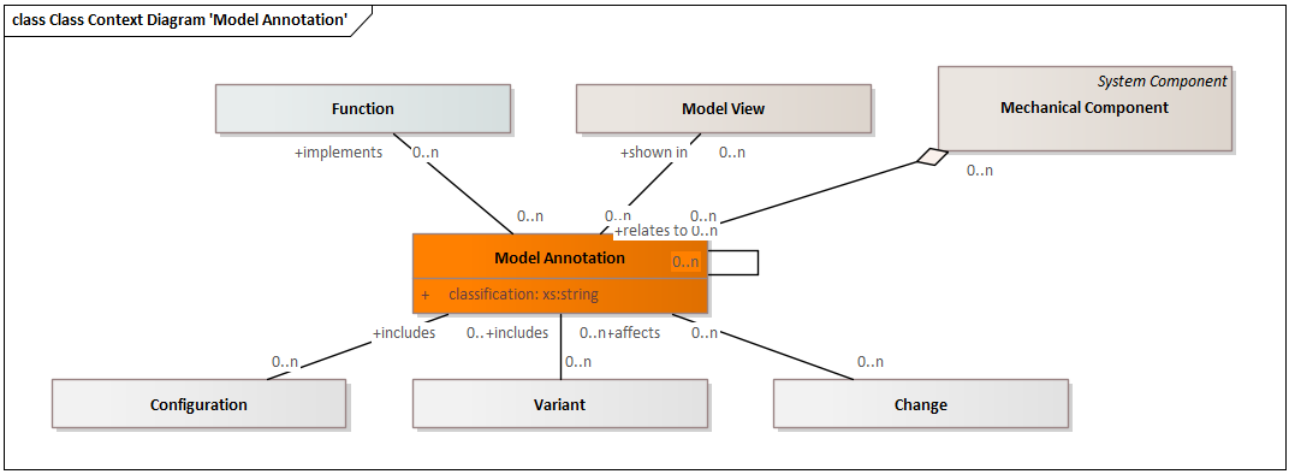


Figure 60 Class Context Diagram 'Model Annotation'

Specializations	Datum Geometric Tolerance Dimensional Size Surface Texture	
Delivered by	Long-Term Archiving	
Delivered to	Mechanical Design	
Attributes		
Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element
Classification		Classification of the annotation
Outbound Relations		
part of	Mechanical Component	Parent mechanical component
shown in	Model View	Parent model view
implements	Function	Product function implemented by the model annotation
relates to	Model Annotation	Related annotations, i. e. datums
Inbound Relations		
Model Annotation	relates to	Related annotations, i. e. datums
Configuration	includes	Entities included in this configuration.
Variant	includes	Entities included in this variant.
Change	affects	Entities affected by this change.

8.5.4.3.2.9. Datum

Represents a tolerancing datum or datum target

Figure 61 Icon

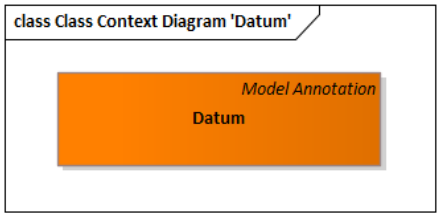


Figure 62 Class Context Diagram 'Datum'

Generalizations	Model Annotation
Delivered by	Plan and prepare Simulation Prepare Testing Inspection Planning Process Planning Cost Calculation Plant Layout Tools & Equipment Design Parts Manufacturing Product Assembly Procurement

8.5.4.3.2.10. Geometric Tolerance

Represents a geometric tolerance, such as form, position, orientation, or runout

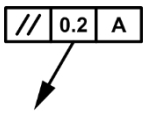


Figure 63 Icon

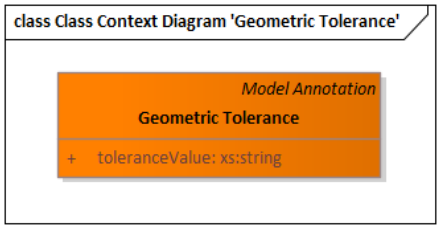


Figure 64 Class Context Diagram 'Geometric Tolerance'

Generalizations	Model Annotation
Related Standards	DIN EN ISO 1101 Tolerierung von Form, Richtung, Ort und Lauf (URL-Dublette!)
Delivered by	Plan and prepare Simulation Prepare Testing Inspection Planning Process Planning Cost Calculation Plant Layout Tools & Equipment Design Parts Manufacturing Product Assembly Procurement

Attributes

Tolerance Value	Tolerance value of a geometric tolerance

8.5.4.3.2.11. Dimensional Size

Represents a size

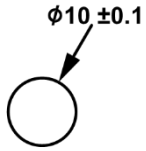


Figure 65 Icon

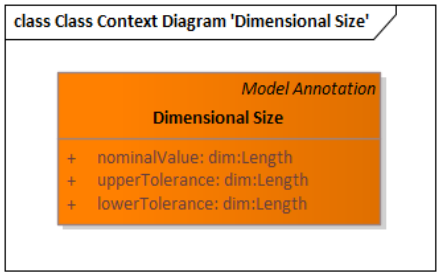


Figure 66 Class Context Diagram 'Dimensional Size'

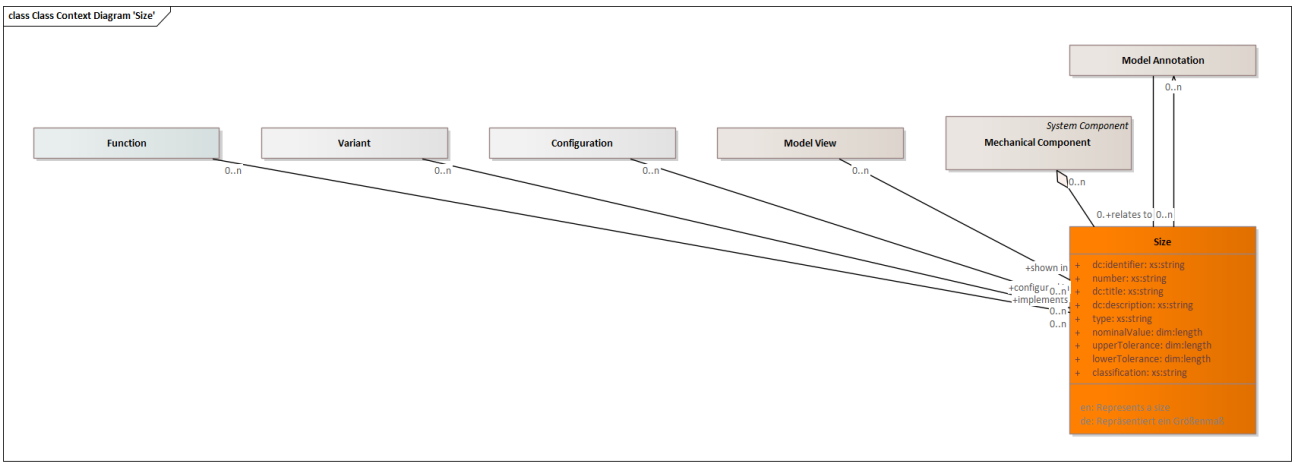


Figure 67 Class Context Diagram 'Size'

Generalizations	Model Annotation
Related Standards	DIN EN ISO 14405 Teil 2: Andere als lineare oder Winkelgrößenmaße (ISO 14405-2:2018) DIN EN ISO 14405 Teil 2: Andere als lineare oder Winkelgrößenmaße (ISO 14405-2:2018) DIN EN ISO 14405 Teil 2: Andere als lineare oder Winkelgrößenmaße (ISO 14405-2:2018)
Delivered by	Plan and prepare Simulation Prepare Testing Inspection Planning Process Planning Cost Calculation Plant Layout Tools & Equipment Design Parts Manufacturing Product Assembly Procurement

Attributes

Nominal Value	Nominal value of the annotation
Upper Tolerance	Upper tolerance of the annotation
Lower Tolerance	Lower tolerance of the annotation

8.5.4.3.2.12. Surface Texture

Represents surface texture such as roughness and waviness

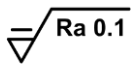


Figure 68 Icon

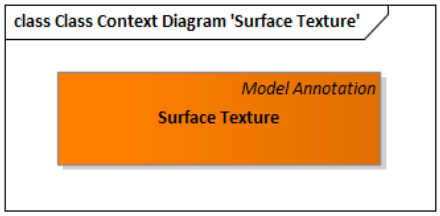


Figure 69 Class Context Diagram 'Surface Texture'

Generalizations	Model Annotation
Related Standards	DIN EN ISO 21920 Teil 1: Angabe der Oberflächenbeschaffenheit (ISO 21920-1:2021) DIN EN ISO 21920 Teil 1: Angabe der Oberflächenbeschaffenheit (ISO 21920-1:2021) DIN EN ISO 21920 Teil 1: Angabe der Oberflächenbeschaffenheit (ISO 21920-1:2021)
Delivered by	Plan and prepare Simulation Prepare Testing Inspection Planning Process Planning Cost Calculation Plant Layout Tools & Equipment Design Parts Manufacturing Product Assembly Procurement

8.5.4.3.2.13. Hole

A cylindrical or conical hole in a part, typically used as a mechanical interface to other mechanical components.

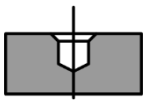


Figure 70 Icon

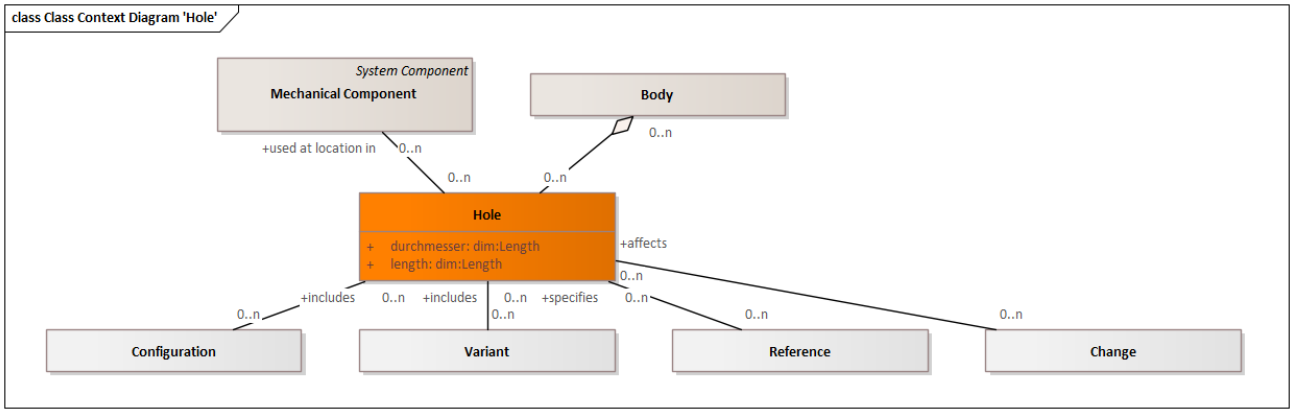


Figure 71 Class Context Diagram 'Hole'

Related Standards	DIN ISO 15786 Vereinfachte Darstellung und Bemaßung von Löchern (ISO 15786:2008)
Delivered by	Plan and prepare Simulation Prepare Testing Disassembly & Recycling Inspection Planning Process Planning Cost Calculation Parts Manufacturing Procurement

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Type	xs:string	Type of an element
Durchmesser		
Length		

Outbound Relations

part of	Body	Parent body
used at location in	Mechanical Component	Context element of a located usage

8.5.4.3.2.14. Gearing

Gearing specification

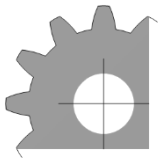


Figure 72 Icon

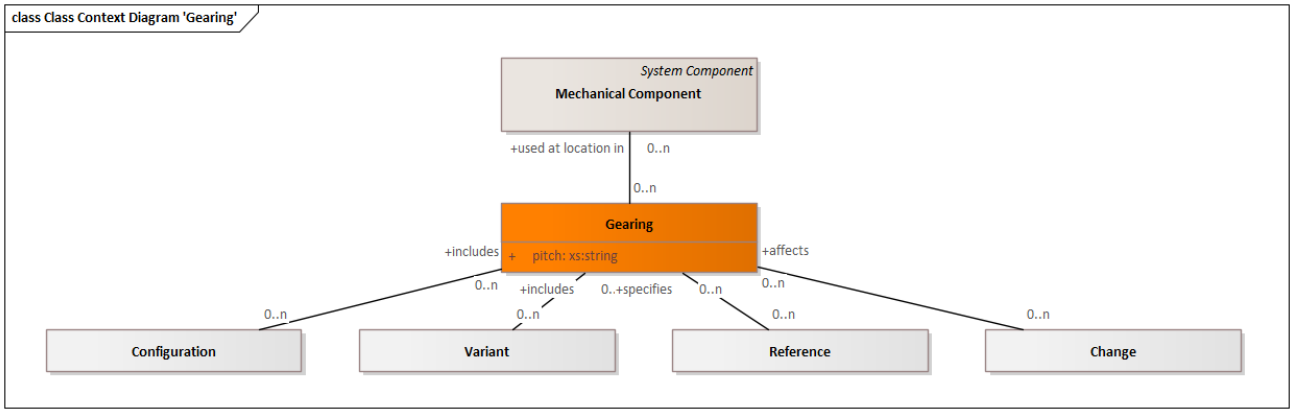


Figure 73 Class Context Diagram 'Gearing'

Related Standards	DIN ISO 15786 Vereinfachte Darstellung und Bemaßung von Löchern (ISO 15786:2008)
Delivered by	Plan and prepare Simulation Prepare Testing Inspection Planning Process Planning Cost Calculation Parts Manufacturing Procurement

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Type	xs:string	Type of an element
Pitch		Pitch of the gearing

Outbound Relations

used at location in	Mechanical Component	Context element of a located usage

8.5.4.3.2.15. Kinematic Mechanism

Represents a kinematic mechanism

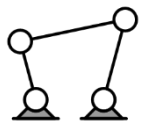


Figure 74 Icon

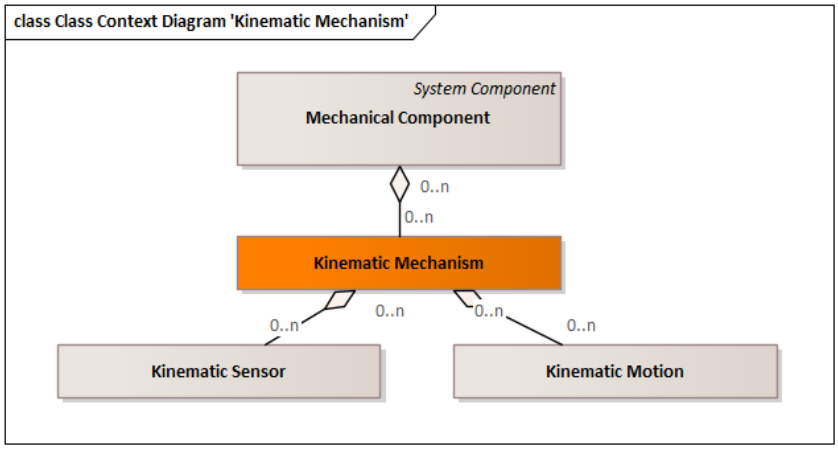


Figure 75 Class Context Diagram 'Kinematic Mechanism'

Delivered by	Long-Term Archiving Plan and prepare Simulation Prepare Testing Inspection Planning Process Planning Cost Calculation Product Assembly
Delivered to	Mechanical Design

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Title	dc:title	Title of the element
Description	dc:description	Description of the element

Outbound Relations

part of	Mechanical Component	Parent mechanical component

8.5.4.3.2.16. Kinematic Joint

Represents a kinematic joint between a pair of component instances

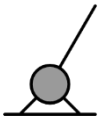


Figure 76 Icon

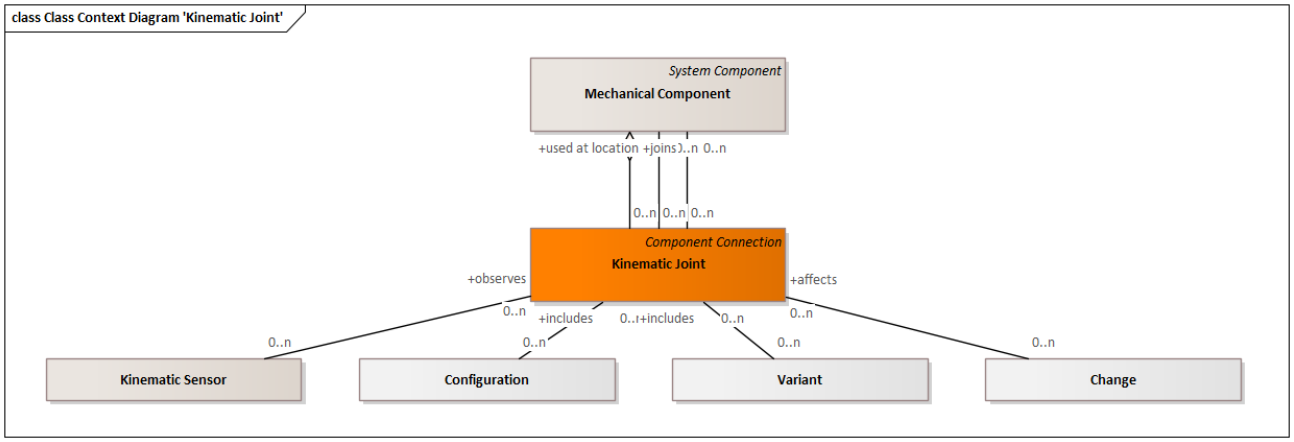


Figure 77 Class Context Diagram 'Kinematic Joint'

Generalizations	Component Connection
Delivered by	Long-Term Archiving Plan and prepare Simulation Prepare Testing Inspection Planning Process Planning Cost Calculation Product Assembly
Delivered to	Mechanical Design

Outbound Relations

part of	Mechanical Component	Parent mechanical component
used at location in	Mechanical Component	Context element of a located usage
joins	Mechanical Component	Represents the system components joined by this component connection.

8.5.4.3.2.17. Kinematic Sensor

Represents a kinematic sensor

Figure 78 Icon

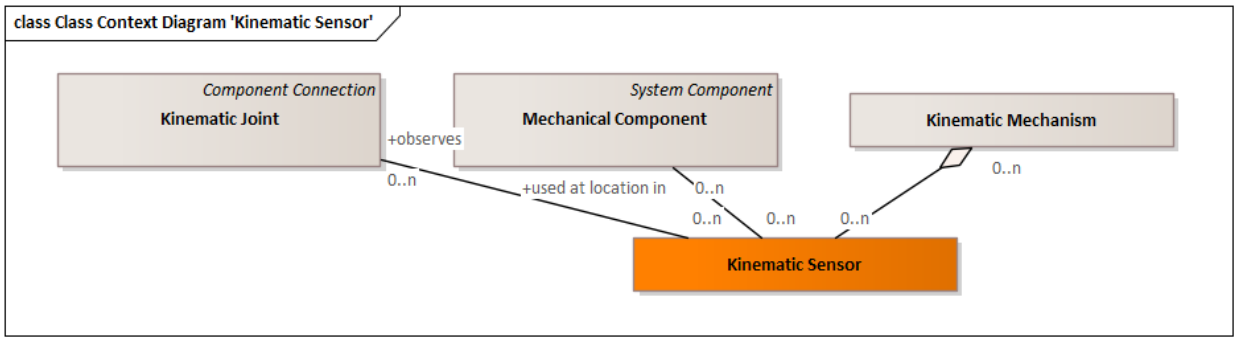


Figure 79 Class Context Diagram 'Kinematic Sensor'

Delivered by	Plan and prepare Simulation

	Prepare Testing	
Delivered to	Mechanical Design	
Attributes		
Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element
Outbound Relations		
part of	Kinematic Mechanism	Parent kinematic mechanism
used at location in	Mechanical Component	Context element of a located usage
observes	Kinematic Joint	Kinematic joint observed by this kinematic sensor

8.5.4.3.2.18. Kinematic Motion

Represents a kinematic motion

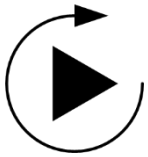


Figure 80 Icon

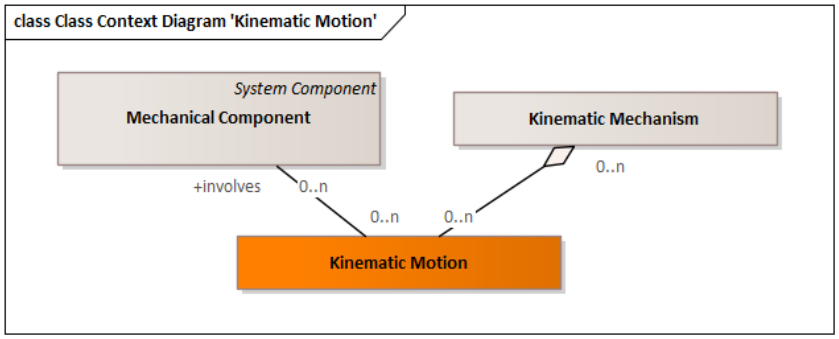


Figure 81 Class Context Diagram 'Kinematic Motion'

Delivered by	Plan and prepare Simulation Prepare Testing	
Delivered to	Mechanical Design	
Attributes		
Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Outbound Relations		
part of	Kinematic Mechanism	Parent kinematic mechanism
involves	Mechanical Component	Components involved by the kinematic motion

8.5.4.3.2.19. Joining Element

Represents a joining element such as a welding spot, a rivet, etc.

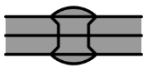


Figure 82 Icon

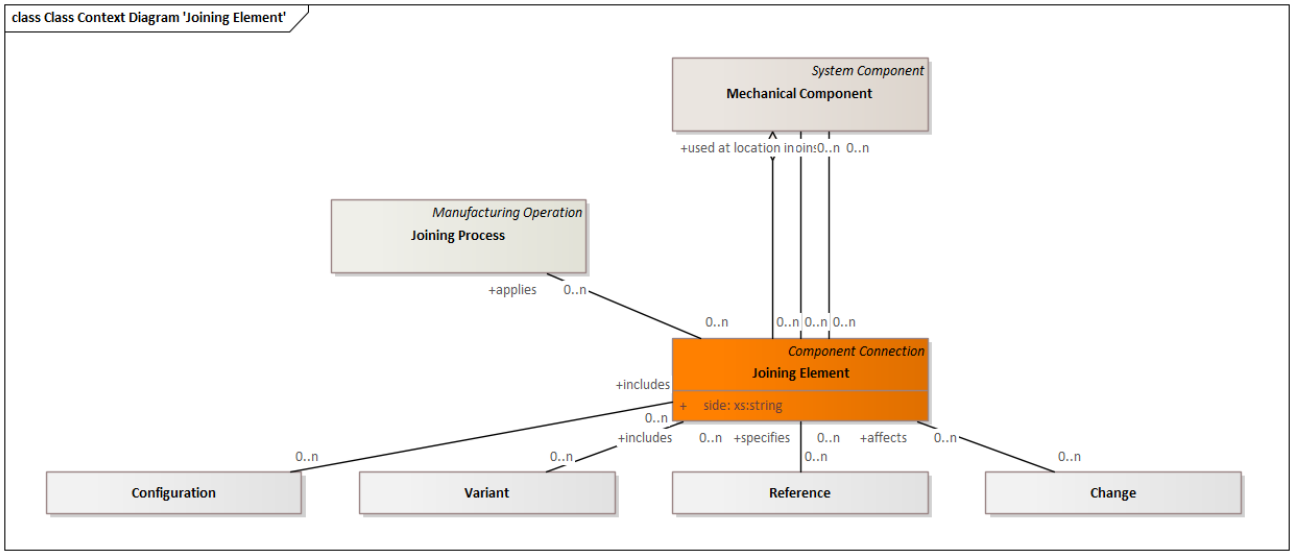


Figure 83 Class Context Diagram 'Joining Element'

Generalizations	Component Connection
Delivered by	Maintenance & Service Long-Term Archiving Plan and prepare Simulation Prepare Testing Disassembly & Recycling Inspection Planning Process Planning Cost Calculation Tools & Equipment Design Product Assembly Quality Inspection Packaging & Logistics Procurement
Delivered to	Mechanical Design Define Joining Elements

Attributes

Side	Side of the joining element

Outbound Relations

part of	Mechanical Component	Parent mechanical component
used at location in	Mechanical Component	Context element of a located usage
joins	Mechanical Component	Represents the system components joined by this component connection.
applies	Joining Process	Joining process used for this joining element

8.5.4.3.2.20. Painting & Coating

Description of a painting or coating for this mechanical component

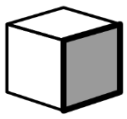


Figure 84 Icon

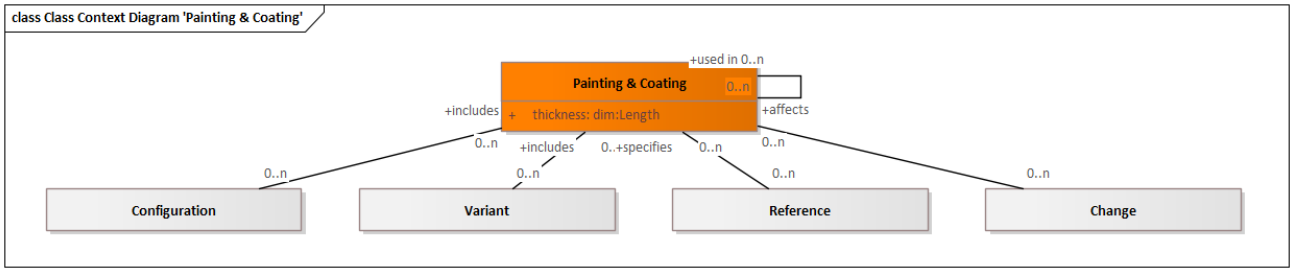


Figure 85 Class Context Diagram 'Painting & Coating'

Delivered by	Plan and prepare Simulation Prepare Testing Disassembly & Recycling Inspection Planning Process Planning Cost Calculation Tools & Equipment Design Parts Manufacturing Procurement

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element
Thickness		Thickness of the painting or coating

Outbound Relations

used in	Painting & Coating	Context element of the usage

Inbound Relations

Painting & Coating	used in	Context element of the usage
Configuration	includes	Entities included in this configuration.
Variant	includes	Entities included in this variant.
Reference	specifies	Entities specified by this reference.
Change	affects	Entities affected by this change.

8.5.4.3.2.21. Marking & Labeling

Figure 86 Icon

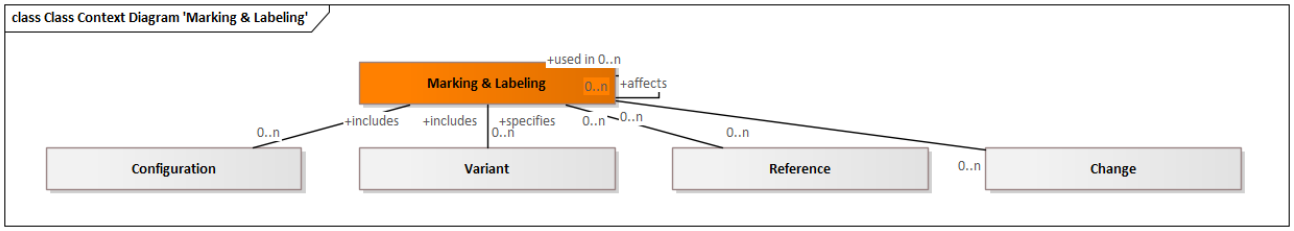


Figure 87 Class Context Diagram 'Marking & Labeling'

Delivered by	Plan and prepare Simulation Prepare Testing Inspection Planning Process Planning Cost Calculation Tools & Equipment Design Parts Manufacturing Procurement

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element

Outbound Relations

used in	Marking & Labeling	Context element of the usage

Inbound Relations

Marking & Labeling	used in	Context element of the usage
Configuration	includes	Entities included in this configuration.
Variant	includes	Entities included in this variant.
Reference	specifies	Entities specified by this reference.
Change	affects	Entities affected by this change.

8.5.4.3.2.22. Heat-treated Area

Heat treatment of a mechanical component

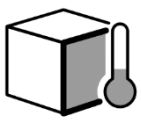


Figure 88 Icon

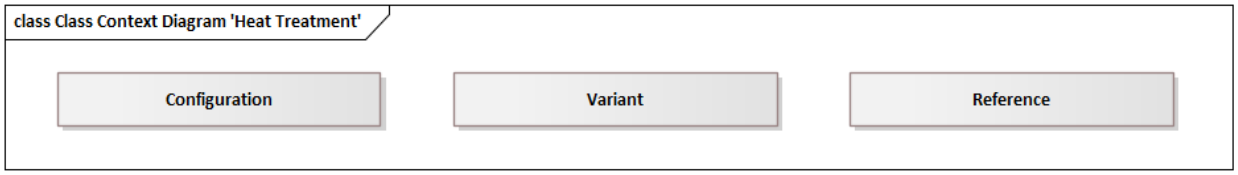


Figure 89 Class Context Diagram 'Heat Treatment'

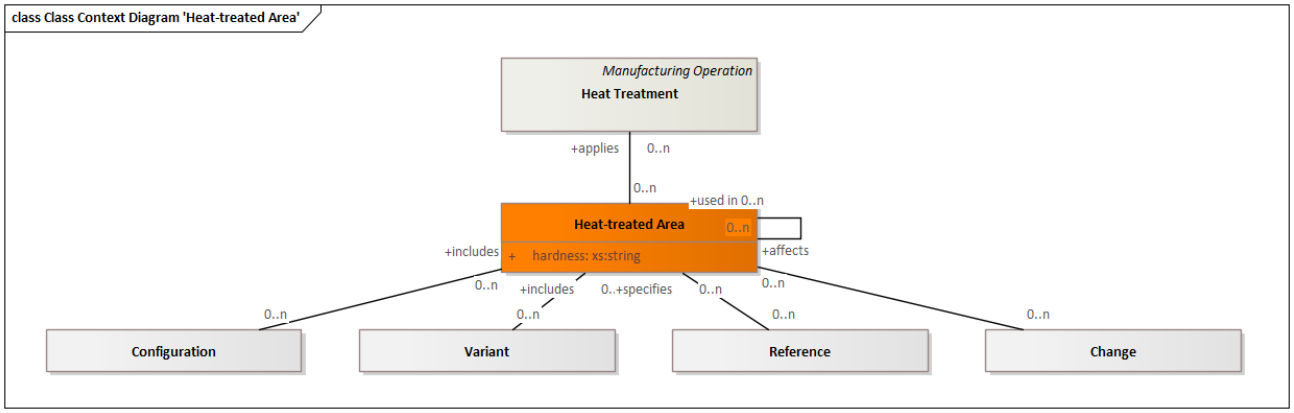


Figure 90 Class Context Diagram 'Heat-treated Area'

Related Standards	DIN EN ISO 4885 Eisenwerkstoffe - Wärmebehandlung - Begriffe
Delivered by	Plan and prepare Simulation Prepare Testing Inspection Planning Process Planning Cost Calculation Tools & Equipment Design Parts Manufacturing Procurement

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element
Hardness		

Outbound Relations

used in	Heat-treated Area	Context element of the usage
applies	Heat Treatment	Heat treatment operation used for this heat treatment specification

Inbound Relations

Heat-treated Area	used in	Context element of the usage
Configuration	includes	Entities included in this configuration.
Variant	includes	Entities included in this variant.
Reference	specifies	Entities specified by this reference.
Change	affects	Entities affected by this change.

8.5.4.3.2.23. Cleanliness

Technical cleanliness of a mechanical component

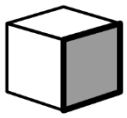


Figure 91 Icon

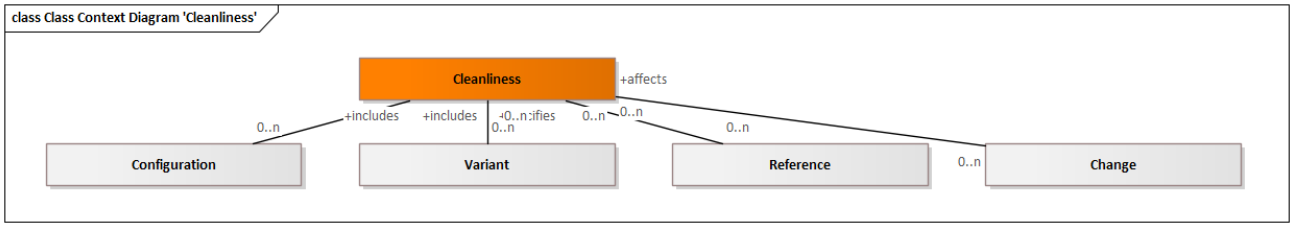


Figure 92 Class Context Diagram 'Cleanliness'

Related Standards	ISO 16232 Cleanliness of components and systems
Delivered by	Plan and prepare Simulation Prepare Testing Inspection Planning Process Planning Cost Calculation Tools & Equipment Design Parts Manufacturing Procurement

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element

8.5.4.3.2.24. Bending Specification

Represents a spatial bending specification that applies a bending angle around an axis.

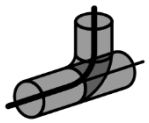


Figure 93 Icon

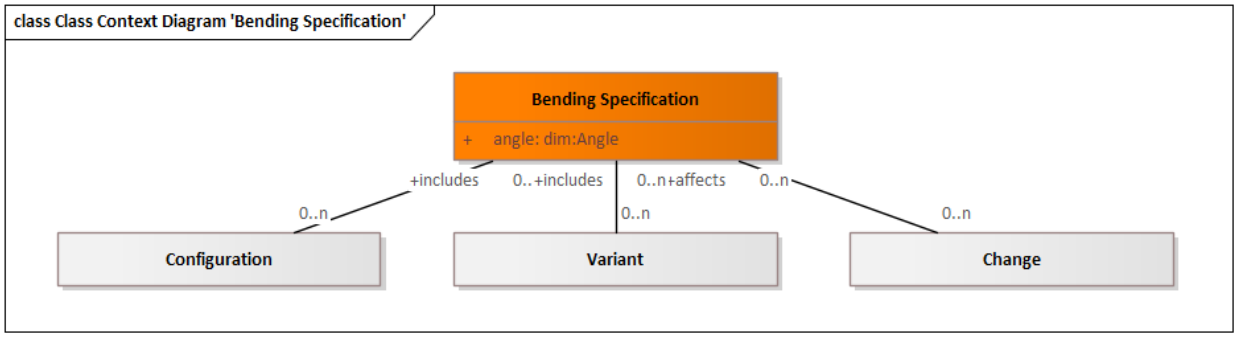


Figure 94 Class Context Diagram 'Bending Specification'

Delivered by	Plan and prepare Simulation Prepare Testing Inspection Planning Process Planning Cost Calculation Tools & Equipment Design Parts Manufacturing Procurement

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Angle		

8.5.4.3.3. Electric/Electronic Design

Includes all entities relevant to electric/electronic design

Figure 95 Icon

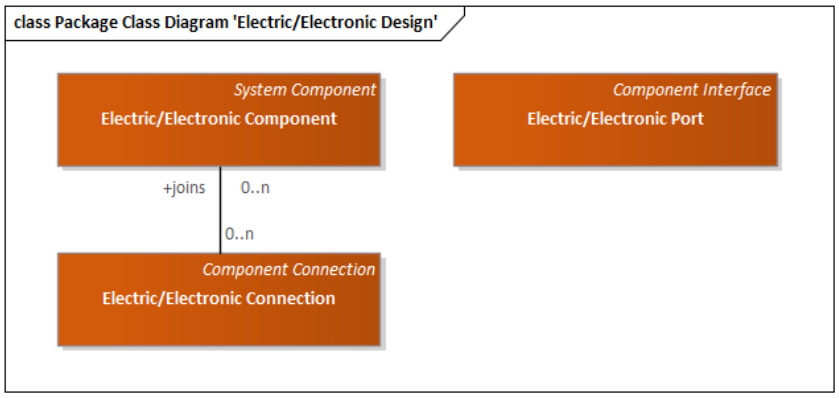


Figure 96 Package Class Diagram 'Electric Electronic Design'

8.5.4.3.3.1. Electric/Electronic Component

Represents an electric/electronic component

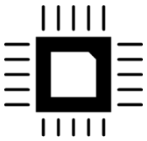


Figure 97 Icon

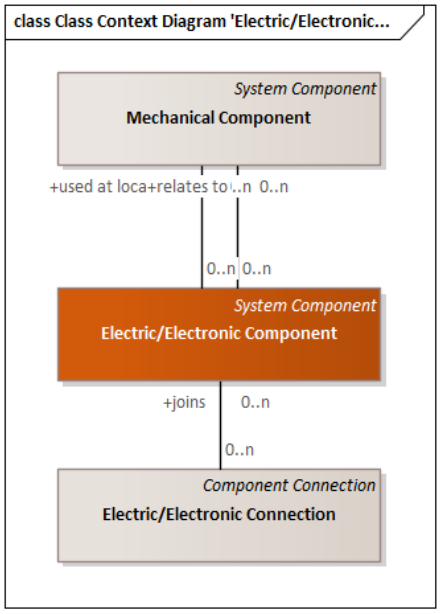


Figure 98 Class Context Diagram 'Electric Electronic Component'

Generalizations	System Component
Delivered by	Product Usage Maintenance & Service Software Engineering Plan and prepare Simulation Prepare Testing Disassembly & Recycling Inspection Planning Electrical/Mechanical Design Collaboration Process Planning Cost Calculation Plant Layout Tools & Equipment Design Procurement
Delivered to	Electrical/Electronical Design

Outbound Relations

used at location in	Mechanical Component	Context element of a located usage
relates to	Mechanical Component	Mechanical component this electric/electronic component relates to

8.5.4.3.3.2. Electric/Electronic Port

Represents an electric/electronic port on an electric/electronic component

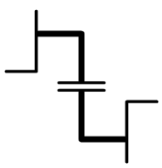


Figure 99 Icon

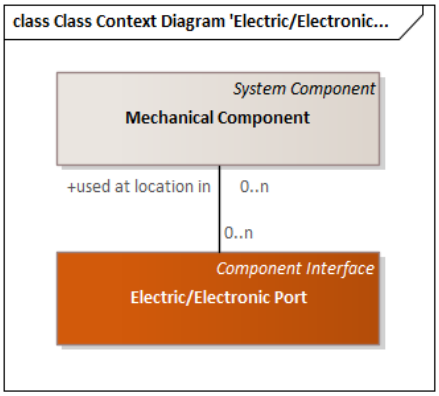


Figure 100 Class Context Diagram 'Electric Electronic Port'

Generalizations	Component Interface
Delivered by	Product Usage Maintenance & Service Software Engineering Plan and prepare Simulation Prepare Testing Disassembly & Recycling Inspection Planning Electrical/Mechanical Design Collaboration Process Planning Cost Calculation Plant Layout Tools & Equipment Design Procurement
Delivered to	Electrical/Electronical Design

Outbound Relations

used at location in	Mechanical Component	Context element of a located usage

8.5.4.3.3.3. Electric/Electronic Connection

Represents an electric/electronic connection

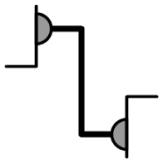


Figure 101 Icon

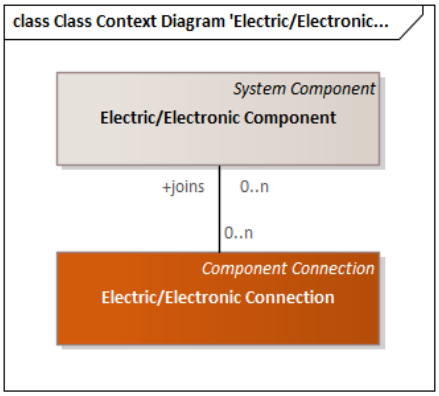


Figure 102 Class Context Diagram 'Electric Electronic Connection'

Generalizations	Component Connection
Delivered by	Product Usage Maintenance & Service Software Engineering Plan and prepare Simulation Prepare Testing Disassembly & Recycling Inspection Planning Electrical/Mechanical Design Collaboration Process Planning Cost Calculation Plant Layout Tools & Equipment Design Procurement
Delivered to	Electrical/Electronical Design

Outbound Relations

joins	Electric/Electronic Component	Represents the system components joined by this component connection.

8.5.4.3.4. Software Design

Includes all entities relevant to software design

Figure 103 Icon

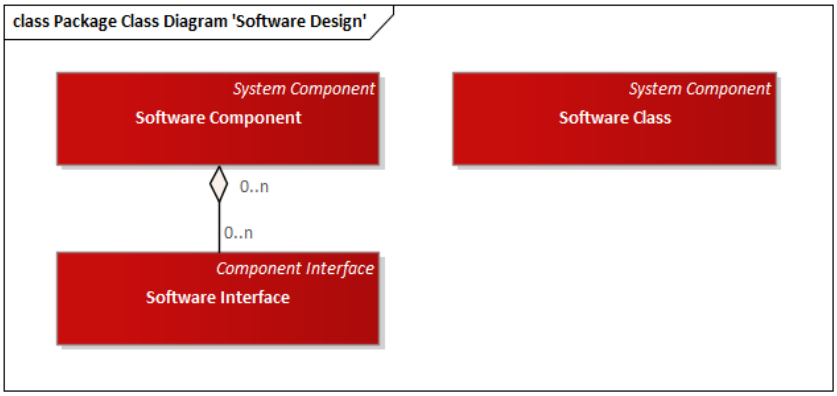


Figure 104 Package Class Diagram 'Software Design'

8.5.4.3.4.1. Software Class

A class in source code that represents a logical collection data and functions

Figure 105 Icon

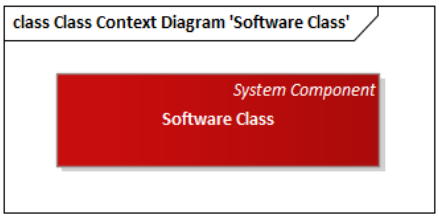


Figure 106 Class Context Diagram 'Software Class'

Generalizations	System Component
Related Standards	ISO/IEC 20246 Bewertungen von Arbeitsergebnissen ISO/IEC 19513 Information technology - Object Management Group Unified Profile for DoDAF and MODAF (UPDM)
Delivered by	Maintenance & Service Plan and prepare Simulation Prepare Testing
Delivered to	Software Engineering

8.5.4.3.4.2. Software Interface

An interface in source code that allows a class to interact with other components of a software

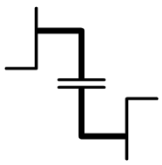


Figure 107 Icon

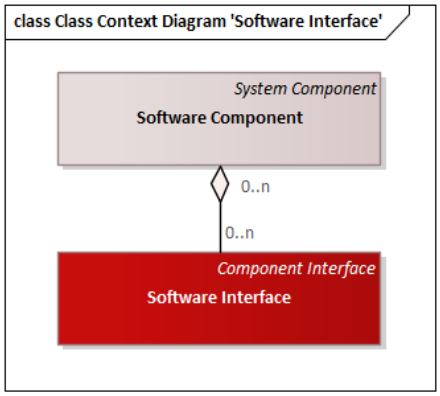


Figure 108 Class Context Diagram 'Software Interface'

Generalizations	Component Interface
Related Standards	ISO/IEC 20246 Bewertungen von Arbeitsergebnissen DO-297 Integrated Modular Avionics (IMA) Development Guidance and Certification Considerations ISO/IEC/IEEE 29119-1 Software and systems engineering - Software testing - Part 1: General concepts
Delivered by	Product Usage Maintenance & Service Plan and prepare Simulation Prepare Testing
Delivered to	Software Engineering

Outbound Relations

part of	Software Component	Parent software component

8.5.4.3.4.3. Software Component

A compiled component of software to be used in a product, i. e. executable or library

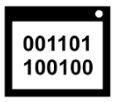


Figure 109 Icon

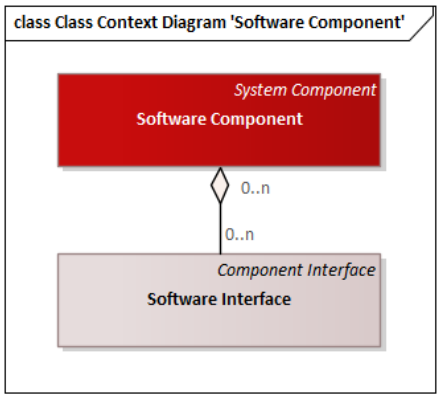


Figure 110 Class Context Diagram 'Software Component'

Generalizations	System Component

Related Standards	ISO/IEC 20246 Bewertungen von Arbeitsergebnissen DO-297 Integrated Modular Avionics (IMA) Development Guidance and Certification Considerations ISO/IEC/IEEE 29119-1 Software and systems engineering - Software testing - Part 1: General concepts
Delivered by	Product Usage Maintenance & Service Plan and prepare Simulation Prepare Testing
Delivered to	Software Engineering

8.5.4.3.5. Fluids

Includes all entities relevant to other domains

Figure 111 Icon

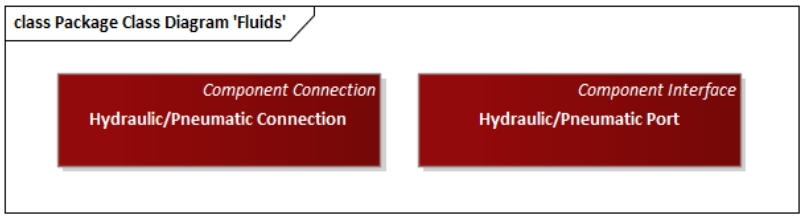


Figure 112 Package Class Diagram 'Fluids'

8.5.4.3.5.1. Hydraulic/Pneumatic Connection

Represents a hydraulic or pneumatic connection, i. e. a tube or hose

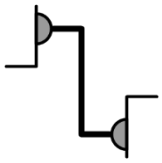


Figure 113 Icon

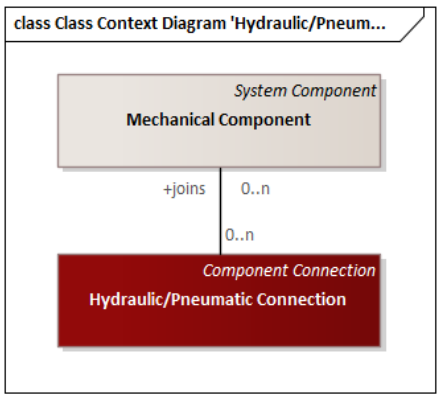


Figure 114 Class Context Diagram 'Hydraulic Pneumatic Connection'

Generalizations	Component Connection

Delivered by	Product Usage Maintenance & Service Plan and prepare Simulation Prepare Testing Disassembly & Recycling Process Planning Packaging & Logistics	
Delivered to	Mechanical Design	

Outbound Relations

joins	Mechanical Component	Represents the system components joined by this component connection.

8.5.4.3.5.2. Hydraulic/Pneumatic Port

Represents a hydraulic or pneumatic port on a mechanical component

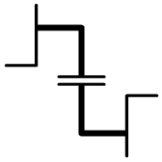


Figure 115 Icon

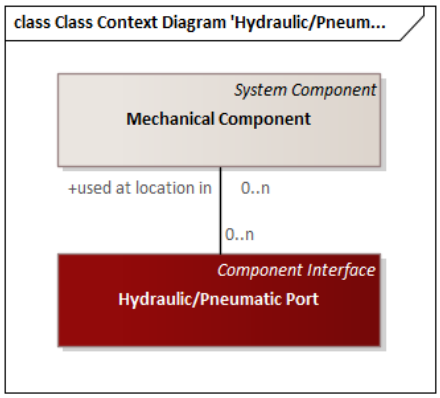


Figure 116 Class Context Diagram 'Hydraulic Pneumatic Port'

Generalizations	Component Interface	
Delivered by	Product Usage Maintenance & Service Plan and prepare Simulation Prepare Testing Disassembly & Recycling Process Planning Packaging & Logistics	
Delivered to	Mechanical Design	

Outbound Relations

used at location in	Mechanical Component	Context element of a located usage

8.5.4.4. Manufacturing & Supply Chain

Includes all entities relevant to manufacturing and supply chain

Figure 117 Icon

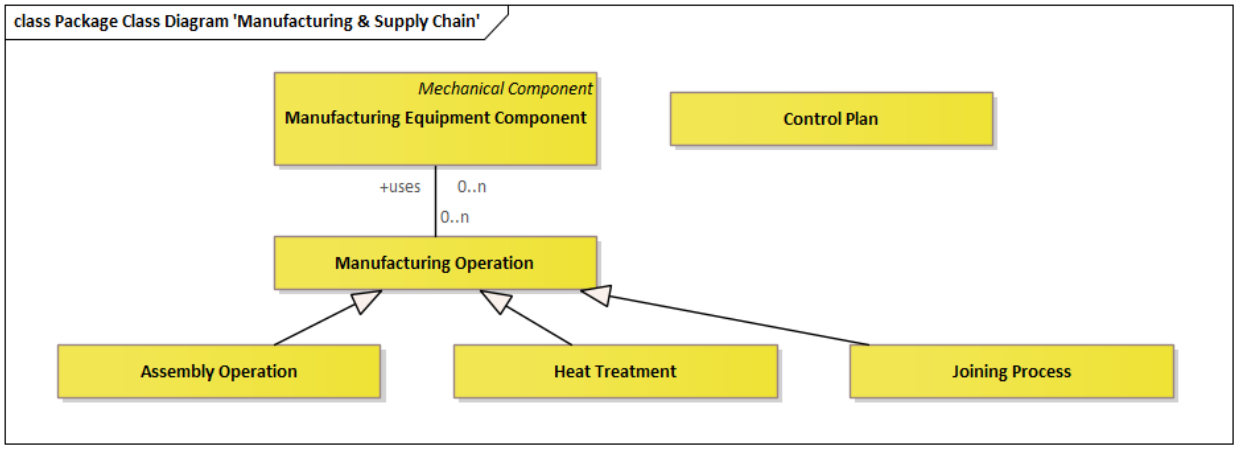


Figure 118 Package Class Diagram 'Manufacturing & Supply Chain'

8.5.4.4.1. Manufacturing Process

Includes all manufacturing process operations

Figure 119 Icon

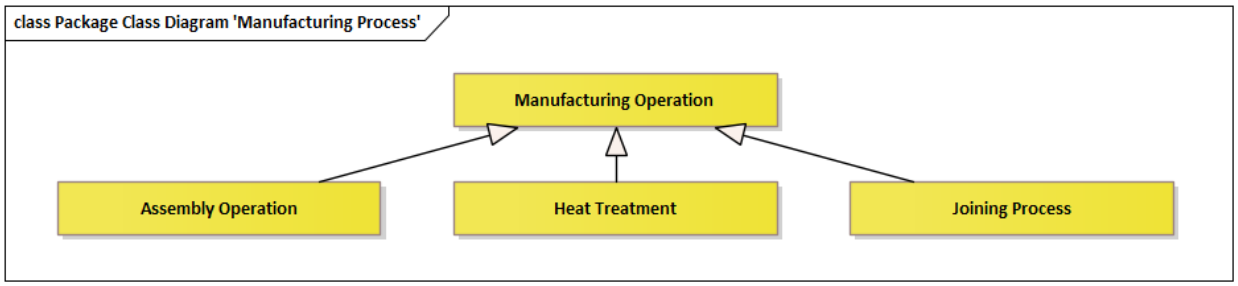


Figure 120 Package Class Diagram 'Manufacturing Process'

8.5.4.4.1.1. Manufacturing Operation

Represents a single or a group of manufacturing operations that lead to a finished, assembled product.

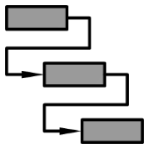


Figure 121 Icon

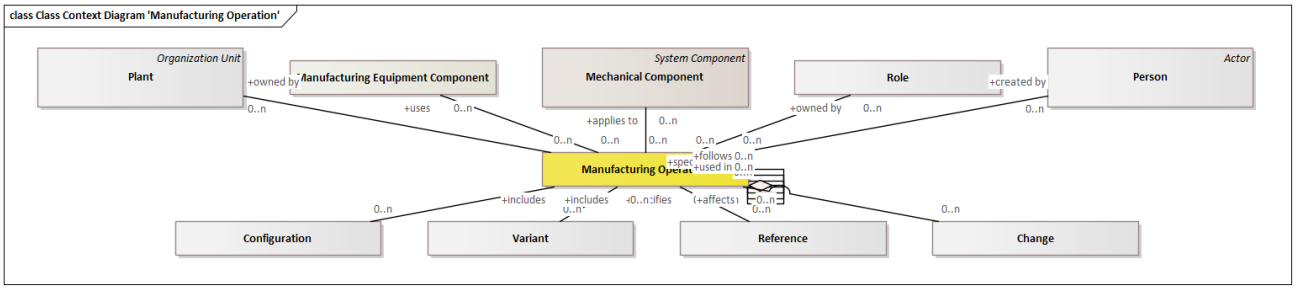


Figure 122 Class Context Diagram 'Manufacturing Operation'

Specializations	Assembly Operation Joining Process Heat Treatment
Delivered by	Cost Calculation Plant Layout Tools & Equipment Design Parts Manufacturing Product Assembly
Delivered to	Process Planning

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element

Outbound Relations

part of	Manufacturing Operation	Parent element
used in	Manufacturing Operation	Context element of the usage
specialism of	Manufacturing Operation	More generic item
created by	Person	The person who created this element
owned by	Role	Role owning this element
applies to	Mechanical Component	Mechanical or electrical component this manufacturing operation applies to
follows	Manufacturing Operation	Preceding manufacturing operation
uses	Manufacturing Equipment Component	Any tools and fixtures used to perform the manufacturing operation.
owned by	Plant	Plant executing the manufacturing operation.

Inbound Relations

Manufacturing Operation	part of	Parent element
Manufacturing Operation	used in	Context element of the usage
Manufacturing Operation	specialism of	More generic item
Manufacturing Operation	follows	Preceding manufacturing operation
Configuration	includes	Entities included in this configuration.
Variant	includes	Entities included in this variant.
Reference	specifies	Entities specified by this reference.
Change	affects	Entities affected by this change.

8.5.4.4.1.2. Assembly Operation

Operation of adding a single part or sub-assembly to an existing single part or sub-assembly

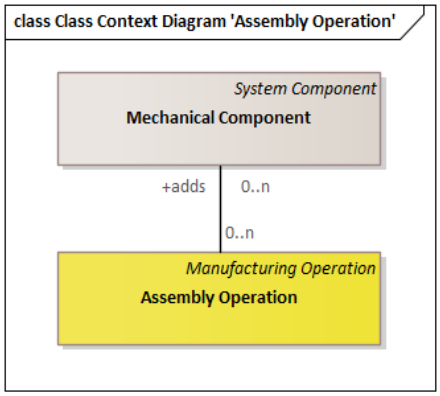


Figure 123 Class Context Diagram 'Assembly Operation'

Generalizations	Manufacturing Operation
Delivered by	Cost Calculation Plant Layout Tools & Equipment Design

Outbound Relations

adds	Mechanical Component	Mechanical or electrical components added by this manufacturing operation

8.5.4.4.1.3. Heat Treatment

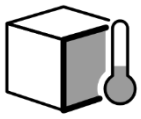


Figure 124 Icon

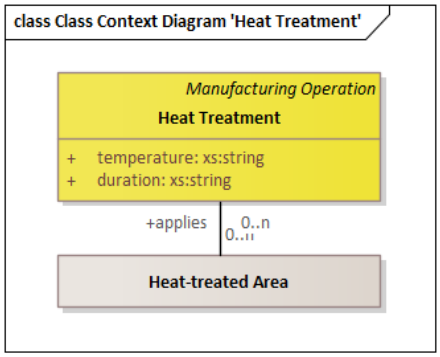


Figure 125 Class Context Diagram 'Heat Treatment'

Generalizations	Manufacturing Operation
Related Standards	DIN 8580 Begriffe, Einteilung
Delivered by	Cost Calculation Plant Layout Tools & Equipment Design

Attributes

Temperature	
Duration	Duration of the heat treatment

8.5.4.4.1.4. Joining Process

Represents a joining process type such as welding, riveting, etc.

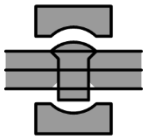


Figure 126 Icon

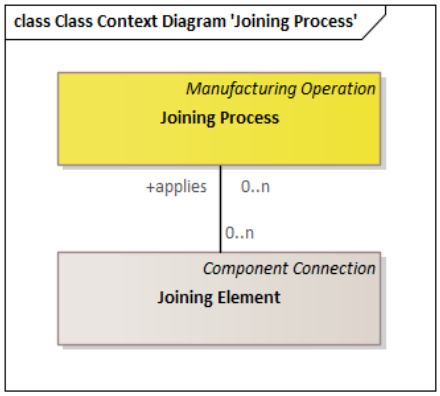


Figure 127 Class Context Diagram 'Joining Process'

Generalizations	Manufacturing Operation
Delivered by	Maintenance & Service Long-Term Archiving Disassembly & Recycling Cost Calculation Plant Layout Tools & Equipment Design Product Assembly Quality Inspection Packaging & Logistics Procurement
Delivered to	Mechanical Design

8.5.4.4.2. Manufacturing Equipment Component

Represents all kinds of tools and equipment used to manufacture parts and assemble the product.

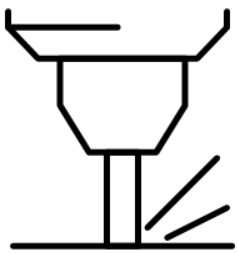


Figure 128 Manufacturing Equipment Component

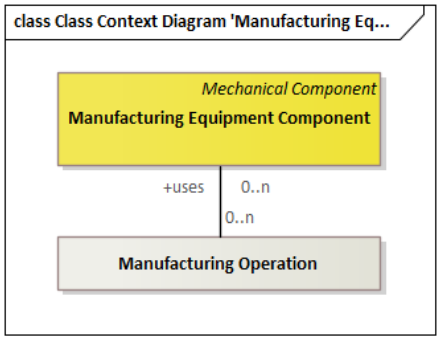


Figure 129 Class Context Diagram 'Manufacturing Equipment Component'

Generalizations	Mechanical Component
Delivered by	Cost Calculation Parts Manufacturing Product Assembly
Delivered to	Plant Layout Tools & Equipment Design

8.5.4.4.3. Control Plan

Defines what verification and validation measures are used from product development through ramp-up to series production.

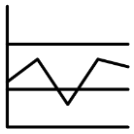


Figure 130 Icon

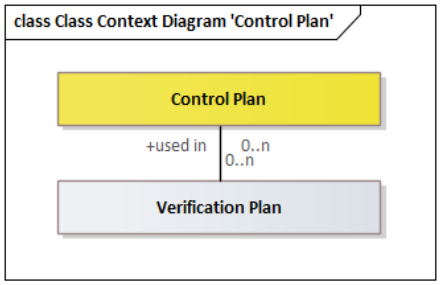


Figure 131 Class Context Diagram 'Control Plan'

Delivered by	Cost Calculation Quality Inspection

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
StartDateTime	xs:dateTime	Start date and time of the element
EndDateTime	xs:dateTime	End date and time of the element

8.5.4.5. Product Verification

Verification of a component or the overall product using virtual (simulation) or physical methods (test, inspection)

Figure 132 Icon

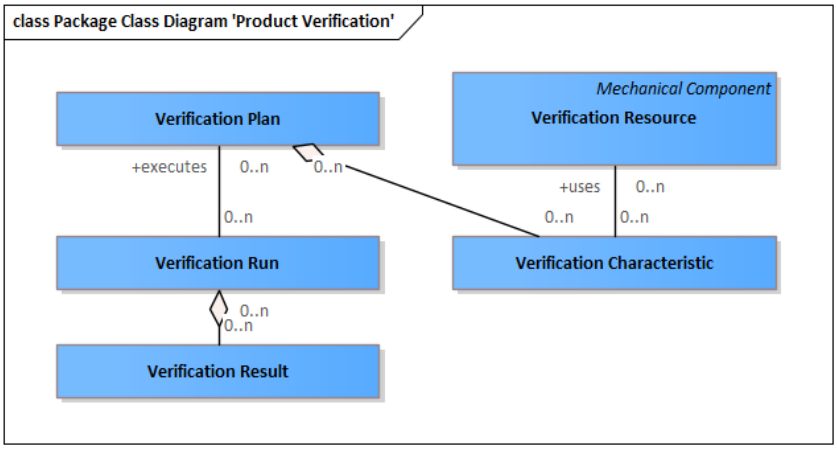


Figure 133 Package Class Diagram 'Product Verification'

8.5.4.5.1. Verification Characteristic

Represents a characteristic that must be inspected to ensure product function.

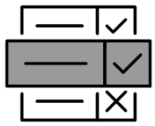


Figure 134 Icon

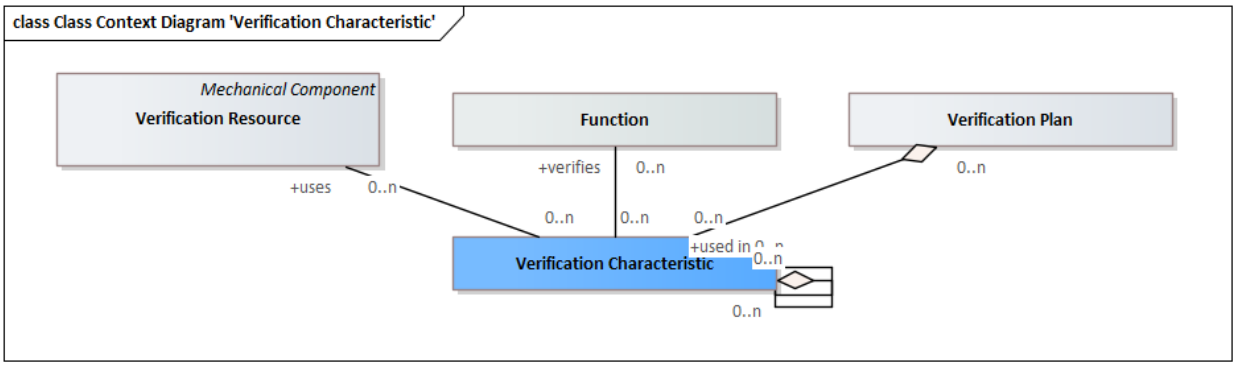


Figure 135 Class Context Diagram 'Verification Characteristic'

Related Standards	ISO 7533 Identification of specifications in the technical product documentation (TPD) VDA 231-300 Digitaler Datenaustausch in der werkstofflichen Bemusterung unter Berücksichtigung der 3D-Daten DIN 23601 Verification - Requirements for measurements to determine the characteristics for size, form, orientation, location and run-out
Delivered by	Quality Inspection
Delivered to	Plan and prepare Simulation

	Prepare Testing Inspection Planning	
Attributes		
Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element
Outbound Relations		
part of	Verification Plan	Parent verification plan this characteristic is allocated to
used in	Verification Characteristic	Context element of the usage
verifies	Function	Function verified by this verification characteristic
uses	Verification Resource	
Inbound Relations		
Verification Characteristic	used in	Context element of the usage

8.5.4.5.2. Verification Plan

Specifies what characteristics must be inspected using which equipment for which purpose.

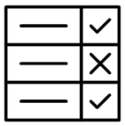


Figure 136 Icon

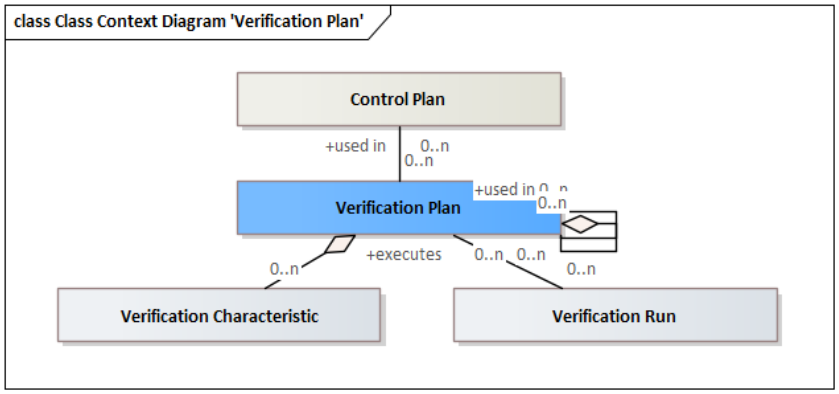


Figure 137 Class Context Diagram 'Verification Plan'

Related Standards	DIN 23601 Verification - Requirements for measurements to determine the characteristics for size, form, orientation, location and run-out
Delivered by	Quality Inspection
Delivered to	Plan and prepare Simulation Prepare Testing Inspection Planning

Attributes		
Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element

Description	dc:description	Description of the element
Type	xs:string	Type of an element
Revision	xs:string	Revisionskennung des Elements
Outbound Relations		
part of	Verification Plan	Parent element
used in	Verification Plan	Context element of the usage
used in	Control Plan	Used in a control plan
Inbound Relations		
Verification Plan	part of	Parent element
Verification Plan	used in	Context element of the usage
Verification Characteristic	part of	Parent verification plan this characteristic is allocated to
Verification Run	executes	Verification plan executed by the verification run

8.5.4.5.3. Verification Resource

Represents a mechanical component used as an inspection resource

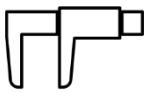


Figure 138 Icon

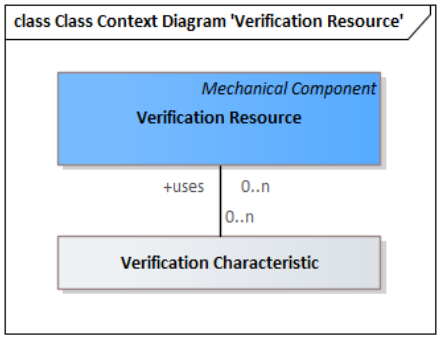


Figure 139 Class Context Diagram 'Verification Resource'

Generalizations	Mechanical Component
Related Standards	DIN 23601 Verification - Requirements for measurements to determine the characteristics for size, form, orientation, location and run-out
Delivered by	Quality Inspection
Delivered to	Plan and prepare Simulation Prepare Testing Inspection Planning

8.5.4.5.4. Verification Run

Represents an inspection run that collects inspection results for each inspection characteristic allocated to an inspection plan.

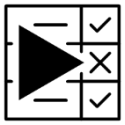


Figure 140 Icon

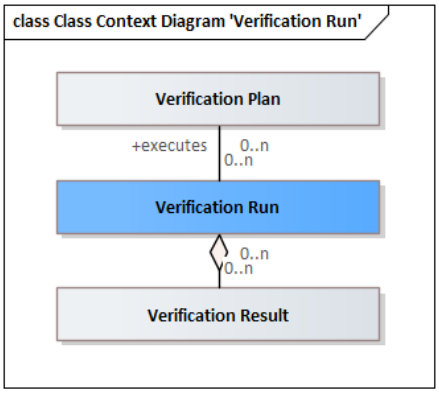


Figure 141 Class Context Diagram 'Verification Run'

Delivered by	Long-Term Archiving
Delivered to	Perform and document Simulation Perform and document Testing

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element
DateTime	xs:dateTime	Date and time of the element

Outbound Relations

owned by	Actor	Actor owning this element
executes	Verification Plan	Verification plan executed by the verification run

8.5.4.5.5. Verification Result

Represents an inspection result for an inspection criterium

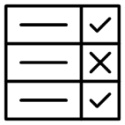


Figure 142 Icon

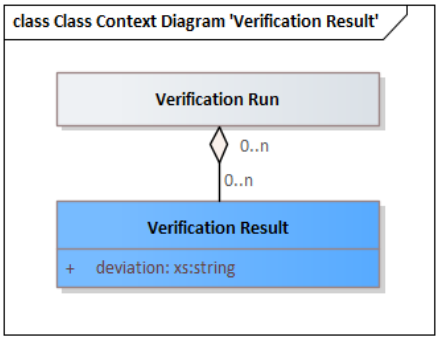


Figure 143 Class Context Diagram 'Verification Result'

Related Standards	ISO 22514 Capability and performance ISO 20170 Decomposition of geometrical characteristics for manufacturing control
Delivered by	Long-Term Archiving
Delivered to	Perform and document Simulation Perform and document Testing

Attributes		
Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element
Deviation		Deviation from specification
Outbound Relations		
part of	Verification Run	Parent verification run

8.5.4.6. Product Validation

Validation of a component or the overall product using virtual (simulation) or physical methods (test)

Figure 144 Icon

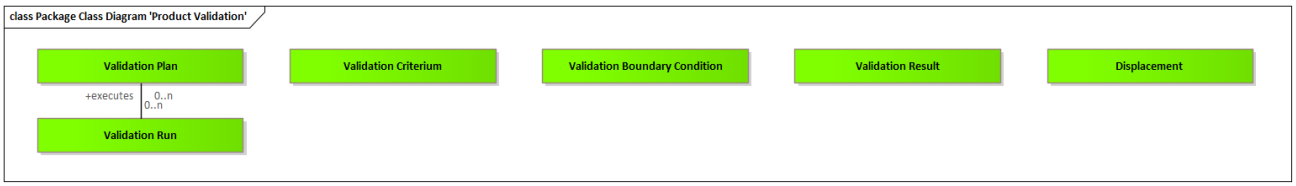


Figure 145 Package Class Diagram 'Product Validation'

8.5.4.6.1. Validation Criterium

A specific validation criterium as part of a validation plan.

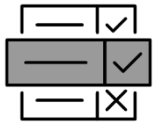


Figure 146 Icon

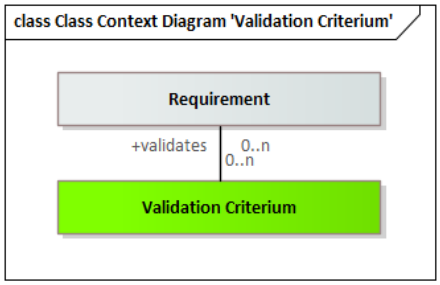


Figure 147 Class Context Diagram 'Validation Criterium'

Related Standards	ISO/IEC/IEEE 29119-1 Software and systems engineering - Software testing - Part 1: General concepts
Delivered by	Perform and document Simulation Perform and document Testing
Delivered to	Plan and prepare Simulation Prepare Testing

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element

Outbound Relations

validates	Requirement	Requirement validated by this validation criterium

8.5.4.6.2. Validation Plan

A specific executable test that examines all aspects including inputs and outputs of a system and then provides a detailed description of the steps that should be taken, the results that should be achieved, and other elements that should be identified. Steps explained in a test case include all details even if they are assumed to be common knowledge. Test cases are used as a technical explanation and reference guide for systems.
https://intranet.prostep.org/display/PROJ/Glossary

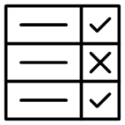


Figure 148 Icon

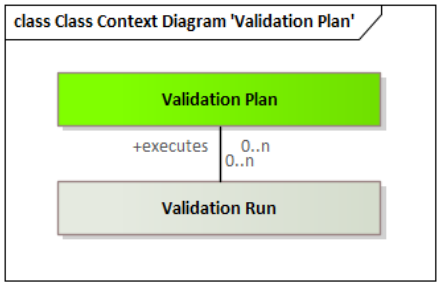


Figure 149 Class Context Diagram 'Validation Plan'

Delivered by	Perform and document Simulation Perform and document Testing
Delivered to	Plan and prepare Simulation Prepare Testing

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element

8.5.4.6.3. Validation Boundary Condition

Validation boundary condition, i. e. force, pressure, temperature or time.

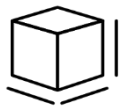


Figure 150 Icon

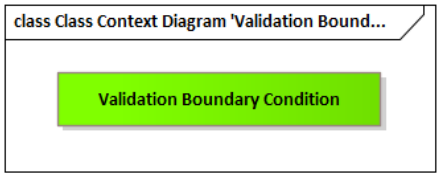


Figure 151 Class Context Diagram 'Validation Boundary Condition'

Delivered by	Perform and document Simulation Perform and document Testing
Delivered to	Plan and prepare Simulation Prepare Testing

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element

8.5.4.6.4. Validation Run

Vorgang einer Validierung

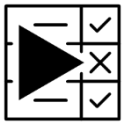


Figure 152 Icon

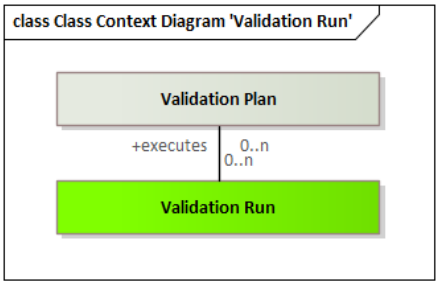


Figure 153 Class Context Diagram 'Validation Run'

Delivered by	Long-Term Archiving Perform and document Simulation Perform and document Testing
Delivered to	Perform and document Simulation Perform and document Testing

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element
DateTime	xs:dateTime	Date and time of the element

Outbound Relations

owned by	Actor	Actor owning this element
executes	Validation Plan	Validation plan executed by the validation run

8.5.4.6.5. Validation Result

Ergebnis einer Validierung

Figure 154 Icon

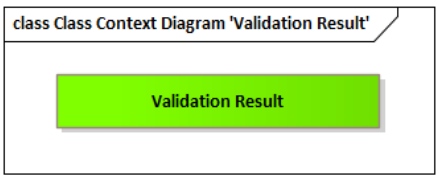


Figure 155 Class Context Diagram 'Validation Result'

Delivered by	Long-Term Archiving

Delivered to	Perform and document Simulation Perform and document Testing	
Attributes		
Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element

8.5.4.6.6. Displacement

Displaced point coordinates

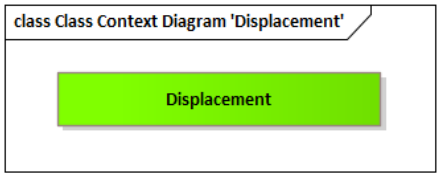


Figure 156 Class Context Diagram 'Displacement'

Delivered by	Long-Term Archiving	
Attributes		
Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number

8.5.4.7. Operation

Includes all entities relevant to product operations

Figure 157 Icon

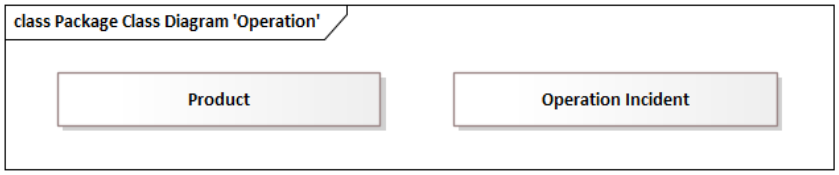


Figure 158 Package Class Diagram 'Operation'

8.5.4.7.1. Product

Represents the commercial view of a product,

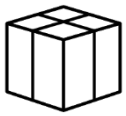


Figure 159 Icon

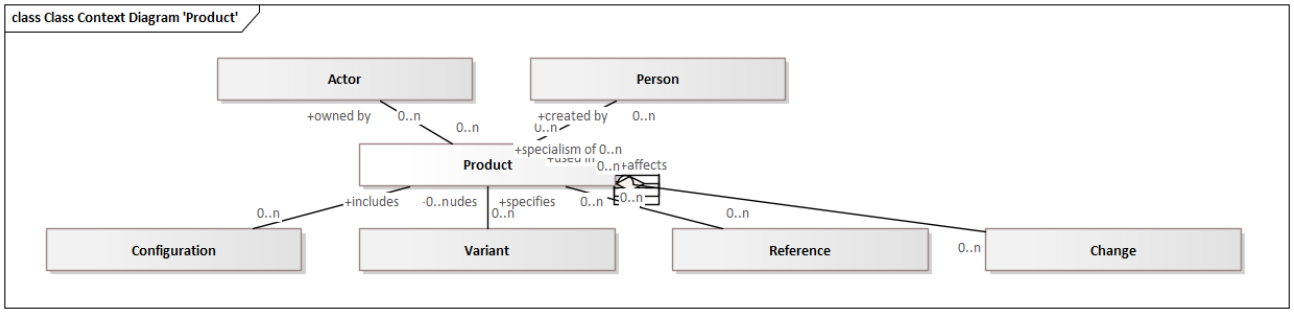


Figure 160 Class Context Diagram 'Product'

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element
StartDateTime	xs:dateTime	Start date and time of the element
EndDateTime	xs:dateTime	End date and time of the element
Revision	xs:string	Revisionskennung des Elements

Outbound Relations

part of	Product	Parent element
used in	Product	Context element of the usage
specialism of	Product	More generic item
created by	Person	The person who created this element
owned by	Actor	Actor owning this element

Inbound Relations

Product	part of	Parent element
Product	used in	Context element of the usage
Product	specialism of	More generic item
Configuration	includes	Entities included in this configuration.
Variant	includes	Entities included in this variant.
Reference	specifies	Entities specified by this reference.
Change	affects	Entities affected by this change.

8.5.4.7.2. Operation Incident

Represents an incident during product operation, i. e. failure, maintenance, etc.

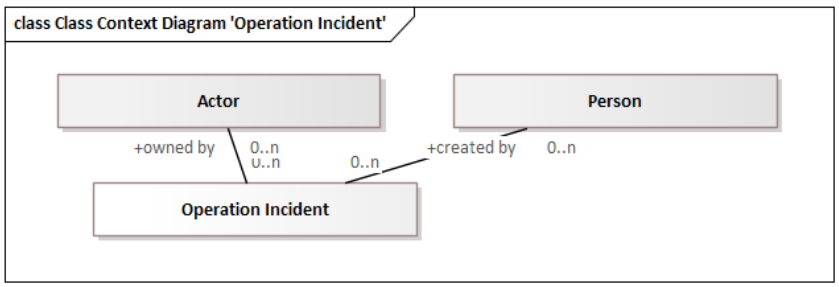


Figure 161 Class Context Diagram 'Operation Incident'

Delivered to	Product Usage

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element
DateTime	xs:dateTime	Date and time of the element

Outbound Relations

created by	Person	The person who created this element
owned by	Actor	Actor owning this element

8.5.4.8. Context Elements

Includes entities that provide context in which related elements are valid, i. e. Revision or Configuration.

Figure 162 Icon

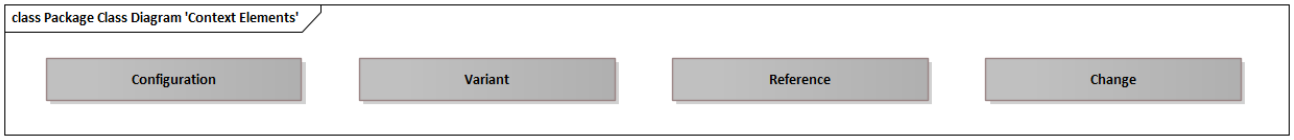


Figure 163 Package Class Diagram 'Context Elements'

8.5.4.8.1. Configuration

Represents a configuration as a context for other elements to be valid. The datamodel allows selected elements to be related to a revision, i. e. mechanical components

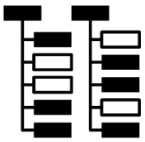


Figure 164 Icon

Figure 165 Class Context Diagram 'Configuration'

Related Standards	ISO 26580 Software and systems engineering - Methods and tools for the feature-based approach to software and systems product line engineering
Delivered to	Mechanical Design Electrical/Electronical Design Software Engineering

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element

StartDateTime	xs:dateTime	Start date and time of the element
EndDateTime	xs:dateTime	End date and time of the element

Outbound Relations

part of	Configuration	Parent element
specialism of	Configuration	More generic item
includes	Use Case	Entities included in this configuration.
includes	Requirement	Entities included in this configuration.
includes	Function	Entities included in this configuration.
includes	Function Failure	Entities included in this configuration.
includes	System Component	Entities included in this configuration.
includes	Component Connection	Entities included in this configuration.
includes	Body	Entities included in this configuration.
includes	Material	Entities included in this configuration.
includes	Model View	Entities included in this configuration.
includes	Model Annotation	Entities included in this configuration.
includes	Hole	Entities included in this configuration.
includes	Gearing	Entities included in this configuration.
includes	Kinematic Joint	Entities included in this configuration.
includes	Joining Element	Entities included in this configuration.
includes	Painting & Coating	Entities included in this configuration.
includes	Marking & Labeling	Entities included in this configuration.
includes	Heat-treated Area	Entities included in this configuration.
includes	Cleanliness	Entities included in this configuration.
includes	Bending Specification	Entities included in this configuration.
includes	Manufacturing Operation	Entities included in this configuration.
includes	Product	Entities included in this configuration.

Inbound Relations

Configuration	part of	Parent element
Configuration	specialism of	More generic item

8.5.4.8.2. Variant

Represents a variant as a context for other elements to be valid. The datamodel allows selected elements to be related to a variant, i. e. mechanical components

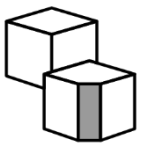


Figure 166 Icon

Figure 167 Class Context Diagram 'Variant'

Related Standards	ISO 26580 Software and systems engineering - Methods and tools for the feature-based approach to software and systems product line engineering
Delivered to	Mechanical Design Electrical/Electronical Design Software Engineering

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element
Outbound Relations		
part of	Variant	Parent element
specialism of	Variant	More generic item
includes	Use Case	Entities included in this variant.
includes	Requirement	Entities included in this variant.
includes	Function	Entities included in this variant.
includes	Function Failure	Entities included in this variant.
includes	System Component	Entities included in this variant.
includes	Component Connection	Entities included in this variant.
includes	Body	Entities included in this variant.
includes	Material	Entities included in this variant.
includes	Model View	Entities included in this variant.
includes	Model Annotation	Entities included in this variant.
includes	Hole	Entities included in this variant.
includes	Gearing	Entities included in this variant.
includes	Kinematic Joint	Entities included in this variant.
includes	Joining Element	Entities included in this variant.
includes	Painting & Coating	Entities included in this variant.
includes	Marking & Labeling	Entities included in this variant.
includes	Heat-treated Area	Entities included in this variant.
includes	Cleanliness	Entities included in this variant.
includes	Bending Specification	Entities included in this variant.
includes	Manufacturing Operation	Entities included in this variant.
includes	Product	Entities included in this variant.
Inbound Relations		
Variant	part of	Parent element
Variant	specialism of	More generic item

8.5.4.8.3. Reference

Related resource

Figure 168 Icon

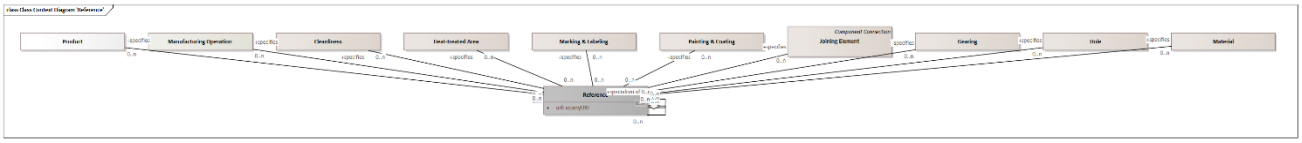


Figure 169 Class Context Diagram 'Reference'

Delivered to	Mechanical Design

	Electrical/Electronical Design Software Engineering	

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element
Revision	xs:string	Revisionskennung des Elements
URL		Unique resource locator of an element

Outbound Relations

part of	Reference	Parent element
specialism of	Reference	More generic item
specifies	Material	Entities specified by this reference.
specifies	Hole	Entities specified by this reference.
specifies	Gearing	Entities specified by this reference.
specifies	Joining Element	Entities specified by this reference.
specifies	Painting & Coating	Entities specified by this reference.
specifies	Marking & Labeling	Entities specified by this reference.
specifies	Heat-treated Area	Entities specified by this reference.
specifies	Cleanliness	Entities specified by this reference.
specifies	Manufacturing Operation	Entities specified by this reference.
specifies	Product	Entities specified by this reference.

Inbound Relations

Reference	part of	Parent element
Reference	specialism of	More generic item

8.5.4.8.4. Change

A change in requirements, engineering, planning or production

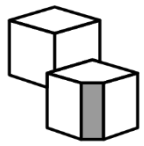


Figure 170 Icon

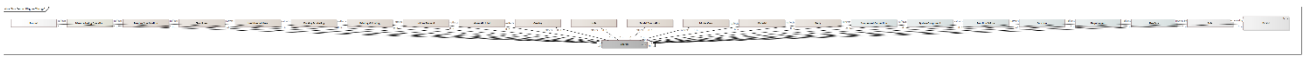


Figure 171 Class Context Diagram 'Change'

Attributes

Identifier	dc:identifier	Unambiguous identifier of the element
Number	xs:string	Legible number or name of the element, i. e. part number
Title	dc:title	Title of the element
Description	dc:description	Description of the element
Type	xs:string	Type of an element
StartDateTime	xs:dateTime	Start date and time of the element
EndDateTime	xs:dateTime	End date and time of the element

Outbound Relations

part of	Change	Parent element
created by	Person	The person who created this element
owned by	Role	Role owning this element
affects	Use Case	Entities affected by this change.
affects	Requirement	Entities affected by this change.
affects	Function	Entities affected by this change.
affects	Function Failure	Entities affected by this change.
affects	System Component	Entities affected by this change.
affects	Component Connection	Entities affected by this change.
affects	Body	Entities affected by this change.
affects	Material	Entities affected by this change.
affects	Model View	Entities affected by this change.
affects	Model Annotation	Entities affected by this change.
affects	Hole	Entities affected by this change.
affects	Gearing	Entities affected by this change.
affects	Kinematic Joint	Entities affected by this change.
affects	Joining Element	Entities affected by this change.
affects	Painting & Coating	Entities affected by this change.
affects	Marking & Labeling	Entities affected by this change.
affects	Heat-treated Area	Entities affected by this change.
affects	Cleanliness	Entities affected by this change.
affects	Bending Specification	Entities affected by this change.
affects	Manufacturing Operation	Entities affected by this change.
affects	Product	Entities affected by this change.

Inbound Relations

Change	part of	Parent element

8.5.5. Use-Cases

Lists all use-cases.

8.5.5.1. CASCaRA

Collaborative Artifact, Specification, Context and Resource Access

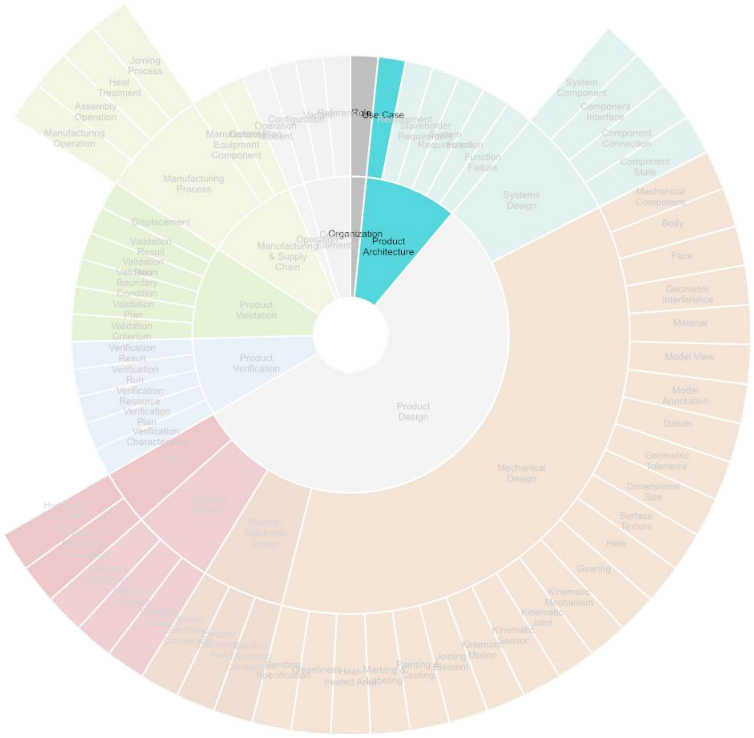


Figure 172 Required Input

Figure 173 Used Tools

8.5.5.1.1. Collaborative Product Requirements Management

Establish and maintain a requirements management process in common with the supplier. Approve requirements specifications ("model based SCD") , get change alerts, manage change requests & releases between OEM and supplier

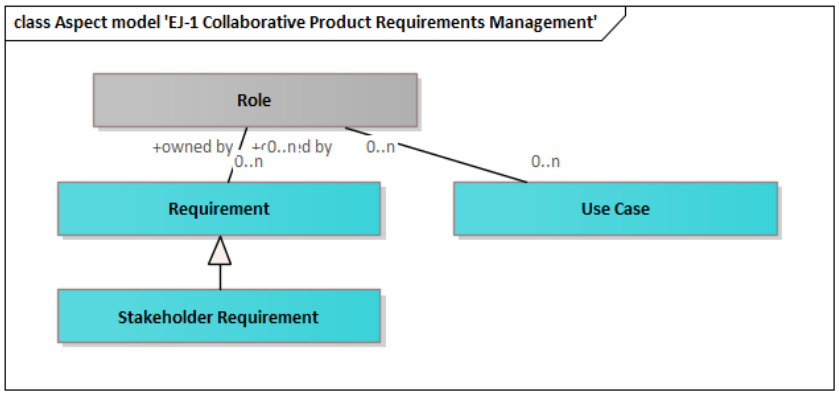


Figure 174 Aspect model 'EJ-1 Collaborative Product Requirements Management'

Inputs	Role Use Case
Outputs	Requirement Stakeholder Requirement

8.5.5.1.2. Collaborative Architecture Design Co-Development

Establish and maintain a method and tool to collaboratively define a system and sub-system architecture & interfaces where supplier provided work products are integrated. Including mission, operational architecture, functional architecture, logical architecture and interface (ICD) definition.

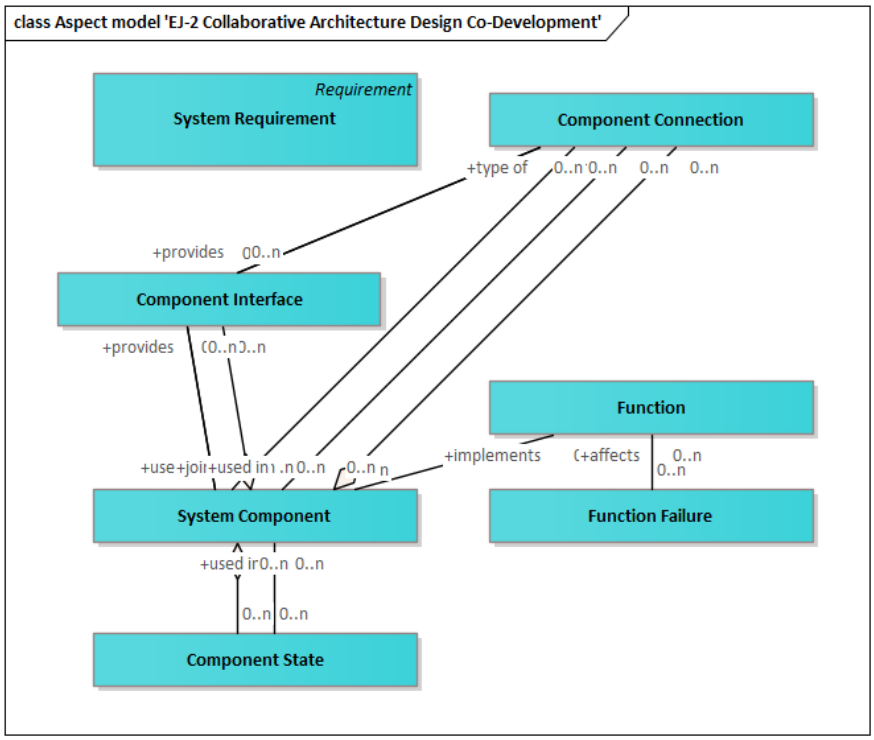


Figure 175 Aspect model 'EJ-2 Collaborative Architecture Design Co-Development'

Outputs	Function Component Interface

	Function Failure System Requirement System Component Component Connection Component State

8.5.5.1.3. Co-Simulation & early Requirements V&V

Establish method and tool to Co-simulate systems, assembling suppliers' sub-systems into a systems integration model, used to perform early design and requirements Verification & Validation. Incorporate Supplier inputs into Safety analysis.

8.5.5.1.4. Collaborative 3D MBD, Electrical and Software Design Co-Development

Establish method and tool to collaboratively develop and manage 3D MBD and E-BOM connected to the Requirements, Functional and Logical Archtecture (L-BOM)

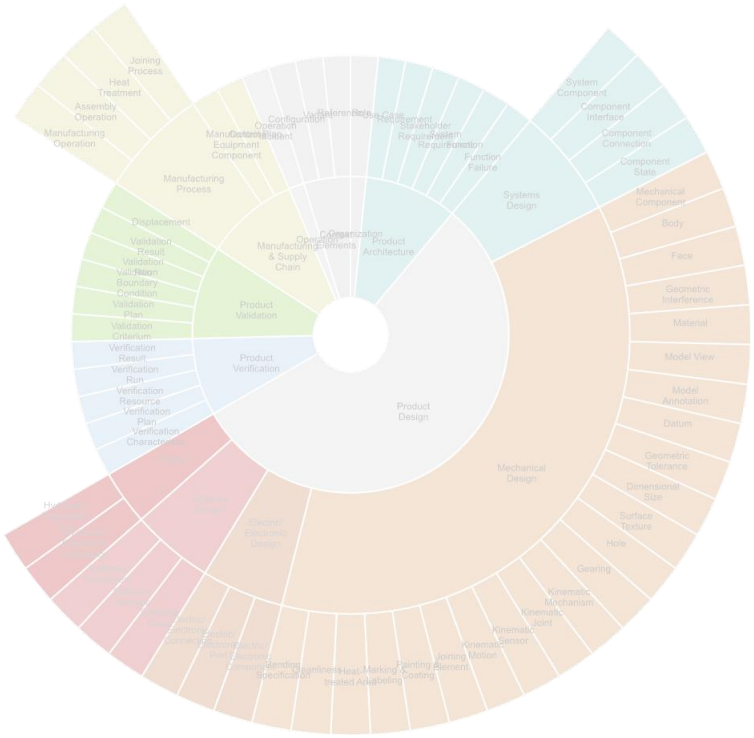


Figure 176 Required Input

Figure 177 Used Tools

8.5.5.1.5. Collaborative Manufacturing Engineering

Establish process and tool for Collabrative Manufacturing MBD: M-BOM, manufacturing process planning and model based work instruction MBI, connected to the 3D MBD

8.5.5.1.6. Collaborative Digital Product Quality Management (APQP)

Establish process and tool for the creation, definition, retention of APQP (Advanced Product Quality Planning) and PPAP (Production part approval process) elements, following indutsry standards such as AS9145.

8.5.5.1.7. LOTAR Long-term Archiving and Retrieval

I want to preserve a baseline of system informationt for later retrieval and reuse. Reference: https://lotar-international.org/

8.5.5.1.8. Model-based Acquisition

Reference: https://www.omg.org/communities/model-based-acquisition-user-community.htm

8.5.5.2. Digital Data Package

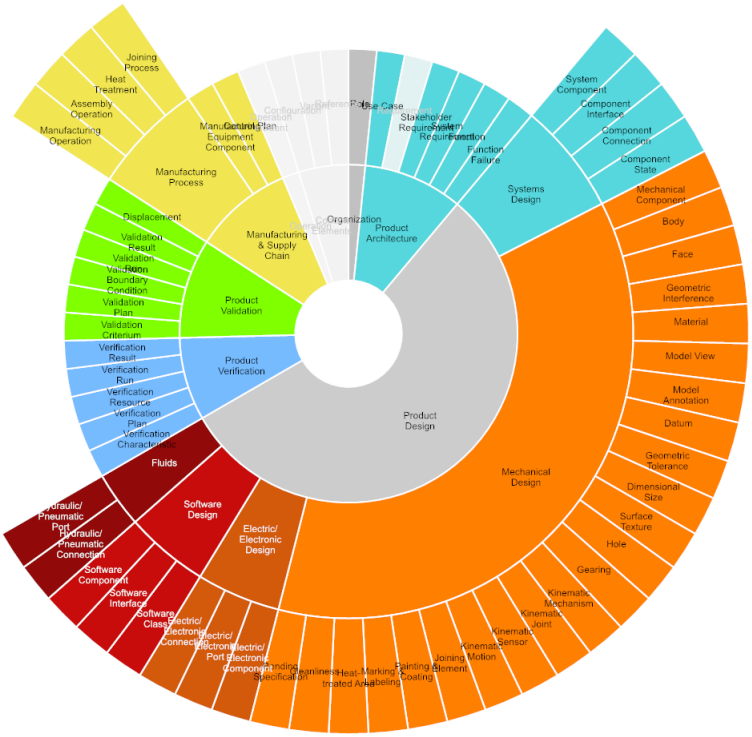


Figure 178 Required Input

Figure 179 Used Tools

8.5.5.2.1. Collaboration Use-Cases

Figure 180 Required Input

Figure 181 Used Tools

8.5.5.2.1.1. Product Development

Product development is the creation of products with new or modified characteristics that offer new or additional benefits to the customer, or comply with new or changed standards, rules and regulations.
Product development may involve modification of an existing product or its presentation or formulation of an entirely new product that satisfies a newly-defined or changed customer need or market segment.
Description based on: https://cio-wiki.org/wiki/Product_Development

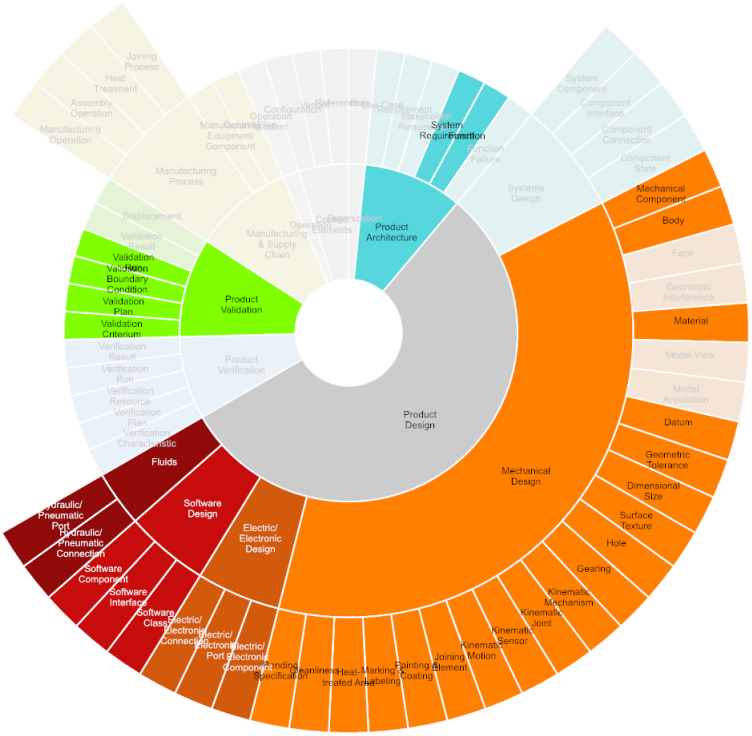


Figure 182 Required Input

8.5.5.2.1.1.1. Concept definition

In conceptual design, new products or systems are defined through requirements and specified through a functional structure in mechanical, electrics/electronics, and software domains. A logical structure specifies the future product architecture in order to implement the specified functions.

Besides new products, existing products may be changed or modified significantly to better comply with customer needs, or to attract new customers. Eventually, the product may be changed or adopted to comply to new standards, regulations or laws.

Conceptual design is usually performed at a high abstraction level and low level of detail, using parameter studies, system models, and rough geometry, if any.

For the Digital Data Package, model-based methods and data for product conceptual design are assumed.

Figure 183 Required Input

8.5.5.2.1.2. Business or Mission Analysis

Product Planning is the ongoing process of identifying and articulating market requirements that define a product's feature set. Product planning serves as the basis for decisions about price, distribution and promotion.
Product planning is the process of creating a product idea and following through on it until the product is introduced to the market. Additionally, a small company must have an exit strategy for its product in case the product does not sell.
Product planning entails managing the product throughout its life using various marketing strategies, including product extensions or improvements, increased distribution, price changes and promotions.
Also includes the identification of relevant legal requirements as well as customs and tax regulations applicable to the product and the respective target markets.
Description based on: https://en.wikipedia.org/wiki/Product_planning

Figure 184 Required Input

Figure 185 Used Tools

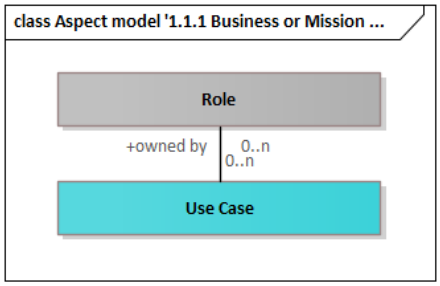


Figure 186 Aspect model '1.1.1 Business or Mission Analysis'

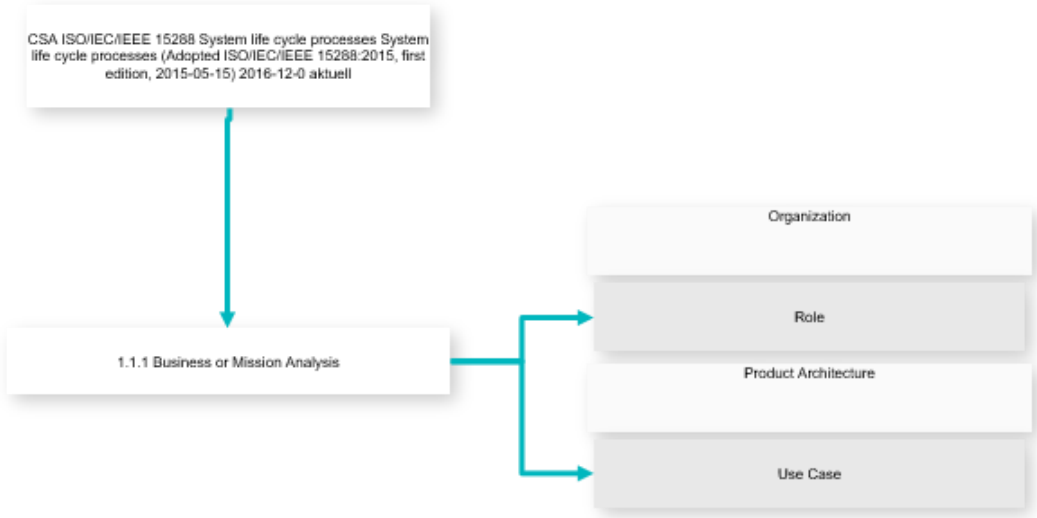


Figure 187 Use-case Diagram

Outputs	Role Use Case

8.5.5.2.1.2.1.1. Requirements Engineering

Requirements engineering is the process of defining, eliciting, documenting, and maintaining requirements in the engineering design process.
Description based on: https://en.wikipedia.org/wiki/Requirements_engineering

Figure 188 Required Input

Figure 189 Used Tools

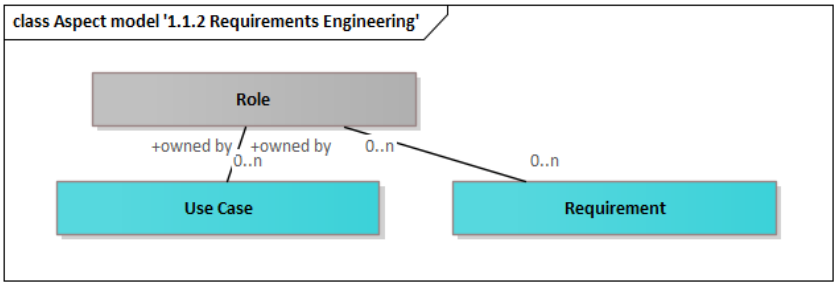


Figure 190 Aspect model '1.1.2 Requirements Engineering'

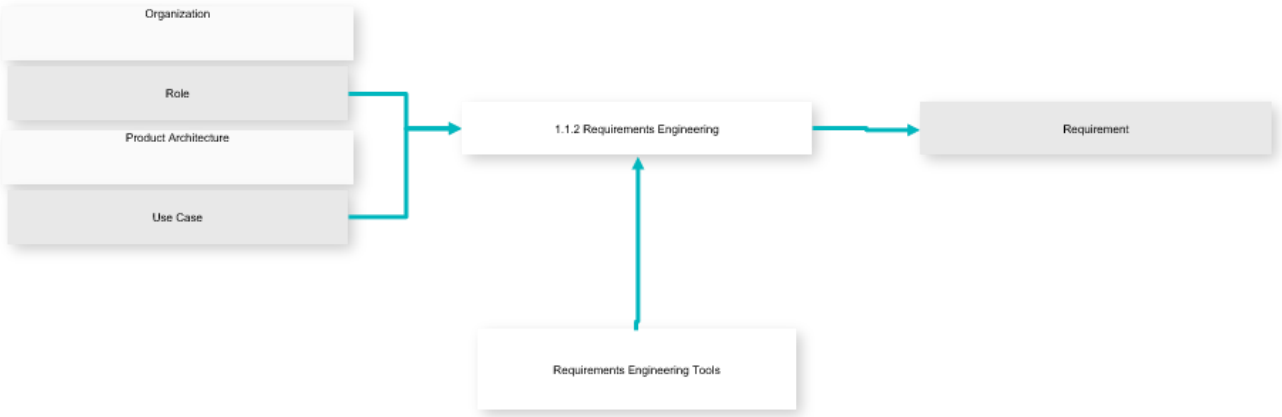


Figure 191 Use-case Diagram

Inputs	Role Use Case
Outputs	Requirement

8.5.5.2.1.3. Systems Architecture

Systems architecture refers to the cross-domain definition of functions and logical structures at systems level, that will be pursued in detail by the individual domains mechanical design, electric/electronic design and software design.

Figure 192 Required Input

Figure 193 Used Tools

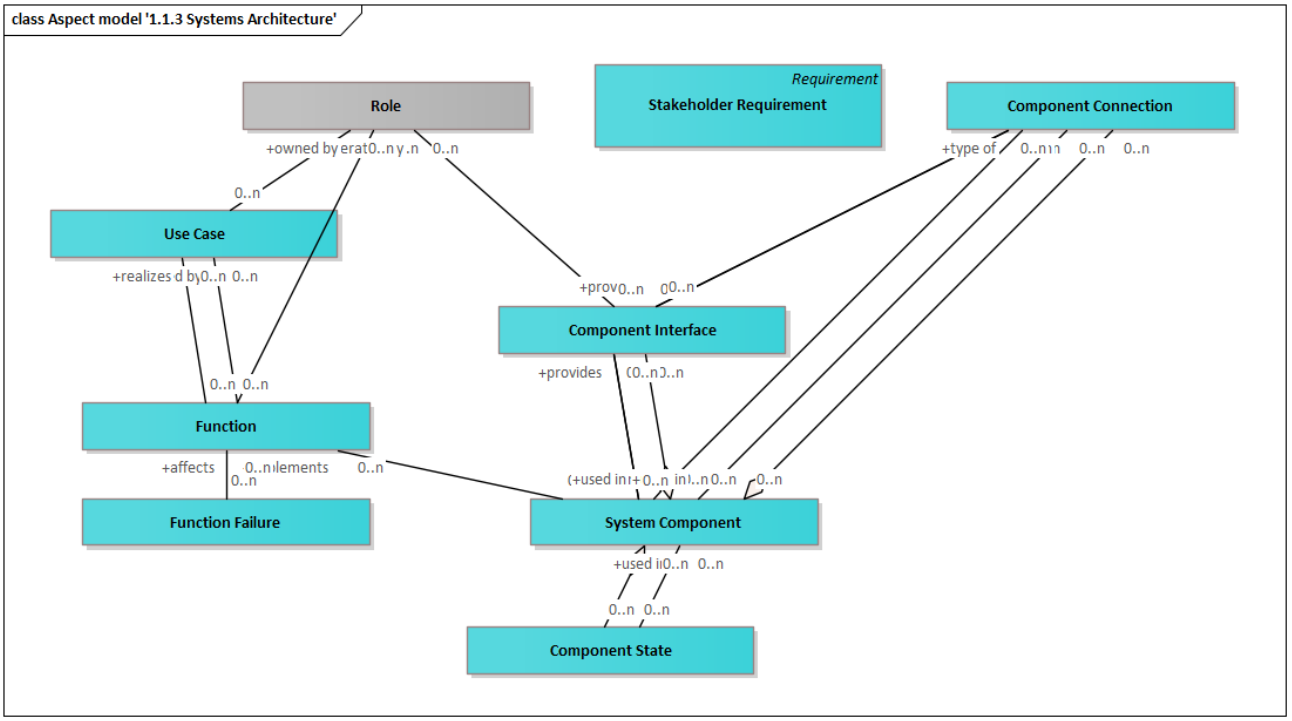


Figure 194 Aspect model '1.1.3 Systems Architecture'

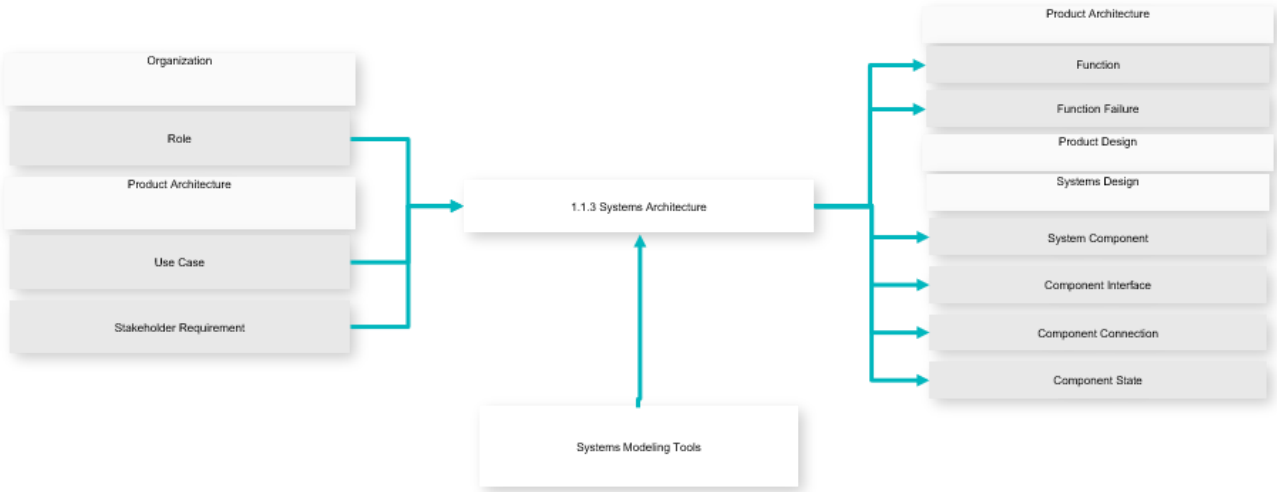


Figure 195 Use-case Diagram

Inputs	Role Use Case Stakeholder Requirement
Outputs	Function Component Interface Function Failure System Component Component Connection Component State

8.5.5.2.1.3.1.1. Functional Safety

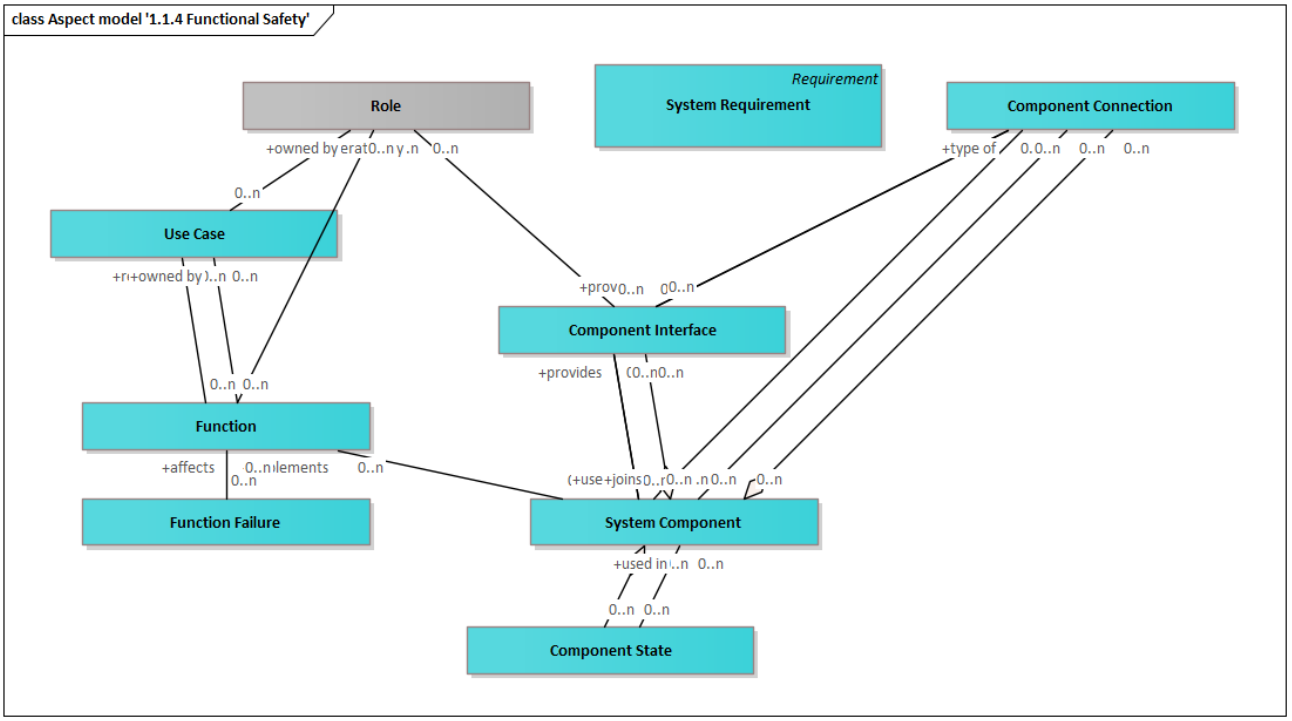


Figure 196 Aspect model '1.1.4 Functional Safety'

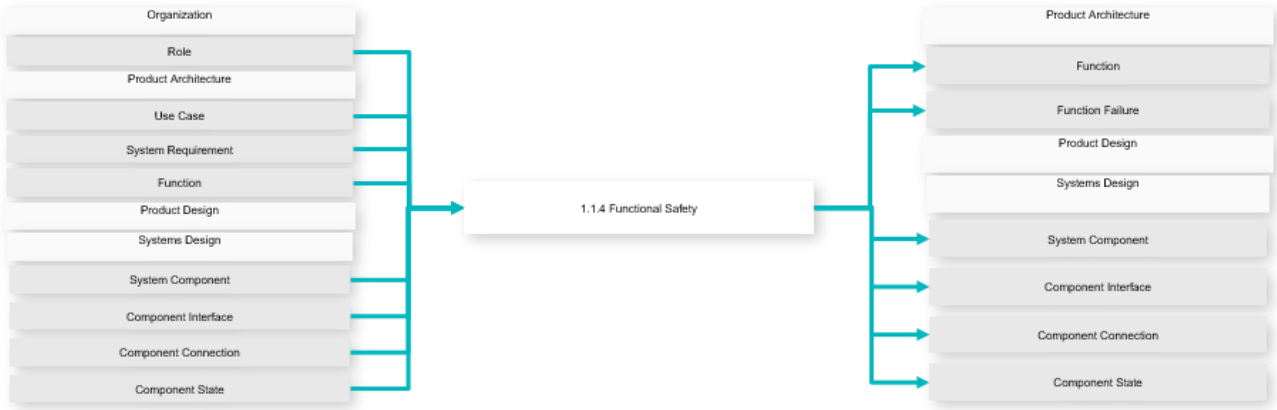


Figure 197 Use-case Diagram

Inputs	Role Use Case Function Component Interface System Requirement System Component Component Connection Component State
Outputs	Function Component Interface Function Failure System Component Component Connection Component State

8.5.5.2.1.3.2. Product Design
see: Product Design

8.5.5.2.1.3.2.1. Mechanical Design

see: Mechanical Design

Inputs	Function Component Interface System Requirement System Component Component Connection Component State
Outputs	Configuration Variant Model Annotation Mechanical Component Hydraulic/Pneumatic Port Kinematic Joint Joining Element Hydraulic/Pneumatic Connection Material Body Geometric Interference Model View Kinematic Mechanism Kinematic Sensor Kinematic Motion Reference Joining Process

8.5.5.2.1.3.2.2. Define Joining Elements

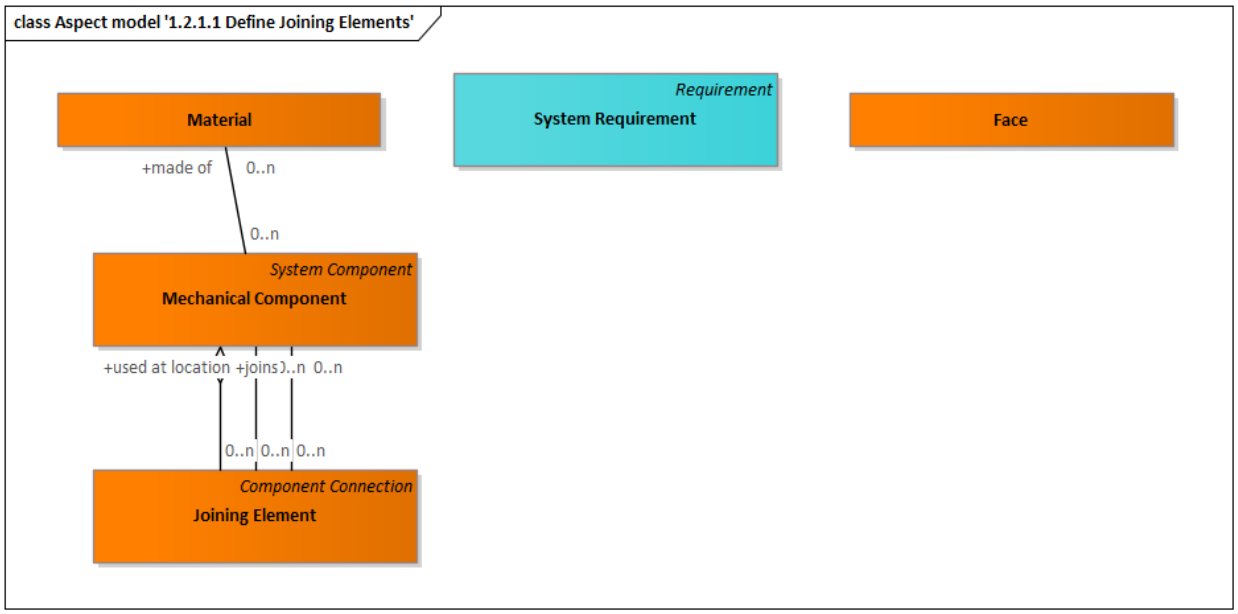


Figure 198 Aspect model '1.2.1.1 Define Joining Elements'

Figure 199 Use-case Diagram

Inputs	System Requirement Mechanical Component Material Face
Outputs	Joining Element

8.5.5.2.1.3.2.3. Electrical/Electronical Design

In Electrical/Electronic Design, all electrical and electronic products functions will be implemented through electrical and electonic components selection and allocation, circuit board design, and wire and harness routing. Electrical/Electronic Design is usually performed using an electrical/electronic CAD system.

Figure 200 Required Input

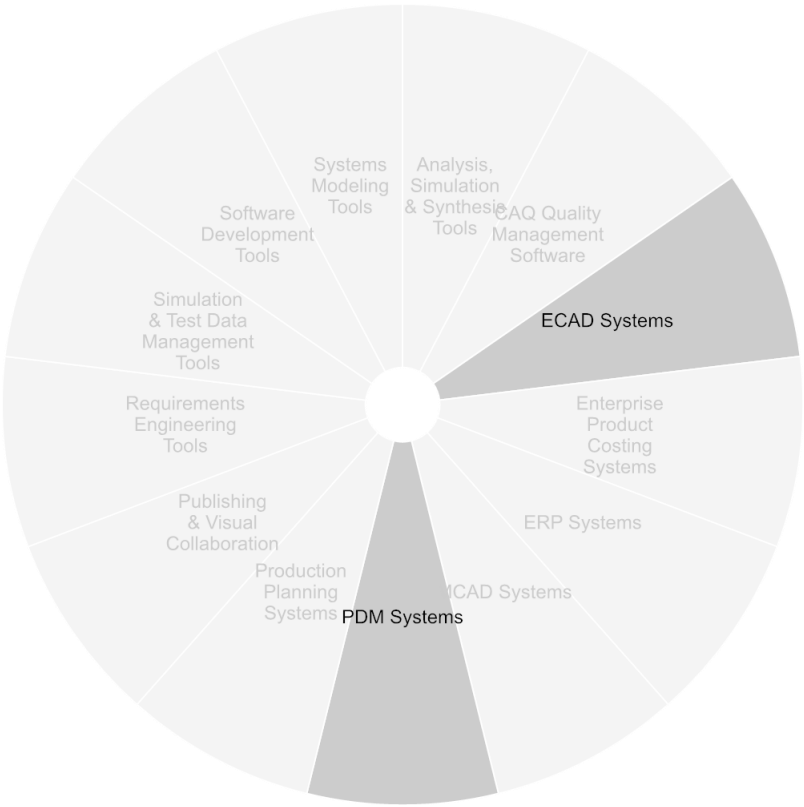


Figure 201 Used Tools

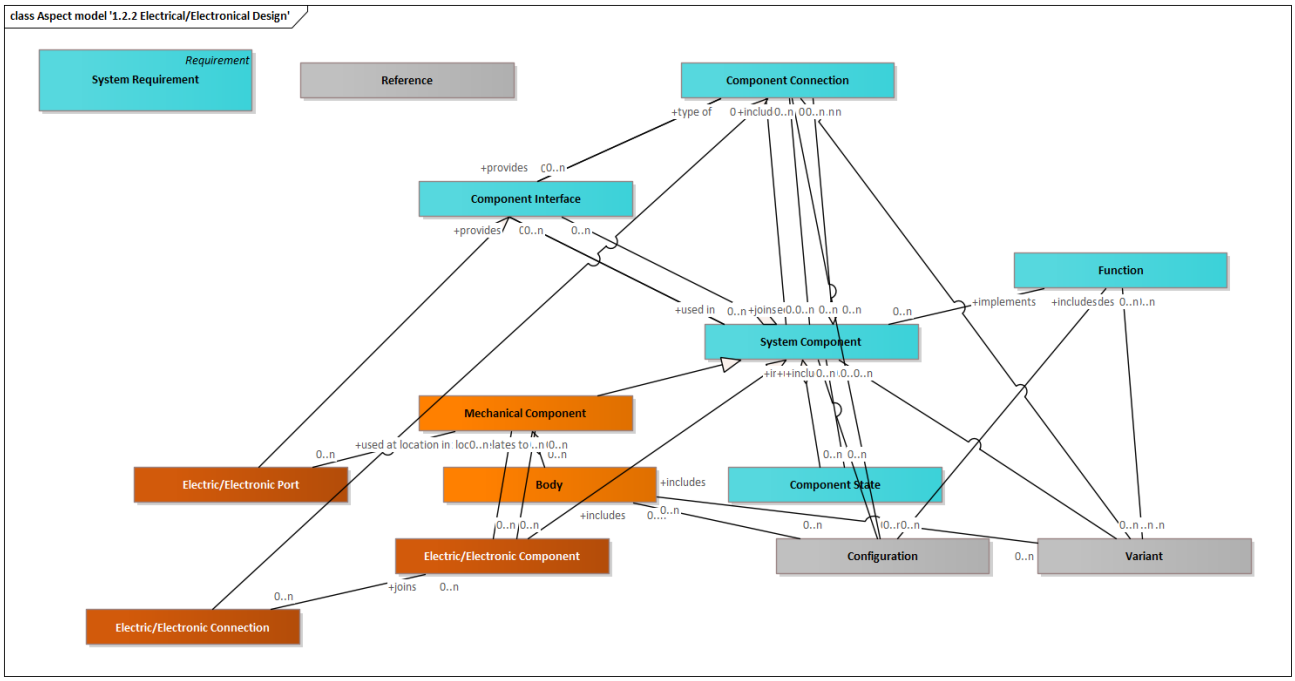


Figure 202 Aspect model '1.2.2 Electrical Electronical Design'

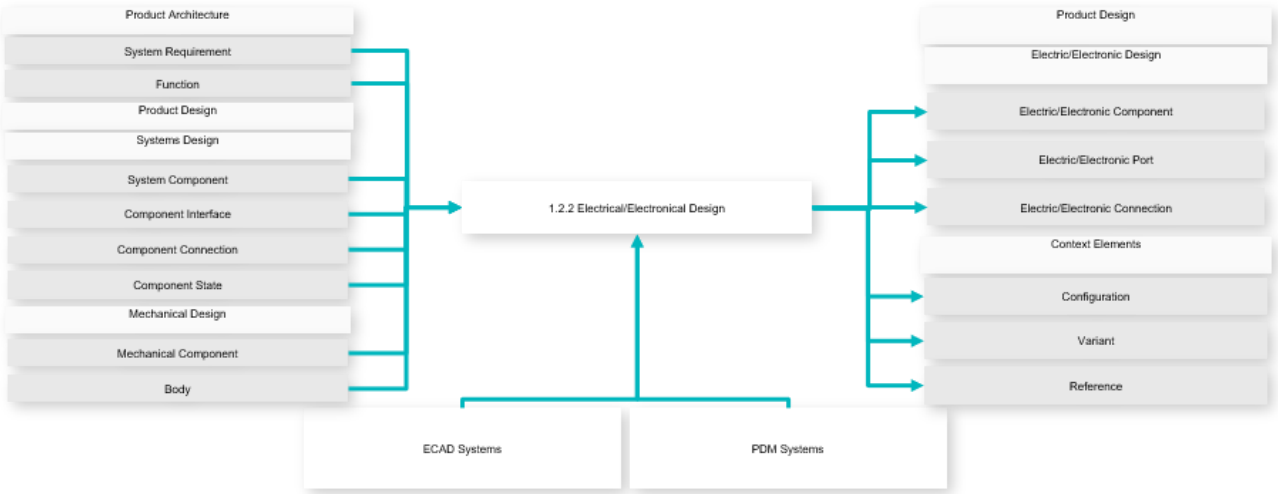


Figure 203 Use-case Diagram

Inputs	Function Component Interface System Requirement System Component Mechanical Component Component Connection Component State Body
Outputs	Configuration Variant Electric/Electronic Component Electric/Electronic Port Electric/Electronic Connection Reference

8.5.5.2.1.3.2.4. Software Engineering

In Software Engineering, embedded software is developed that will connect the product to the user, as well as to other products and systems.
For configuration reasons it may be that software is treated a part to be considered in Bills of Material of products or systems and to be configured related to mechanical or electronic configurations of a system.

Figure 204 Required Input

Figure 205 Used Tools

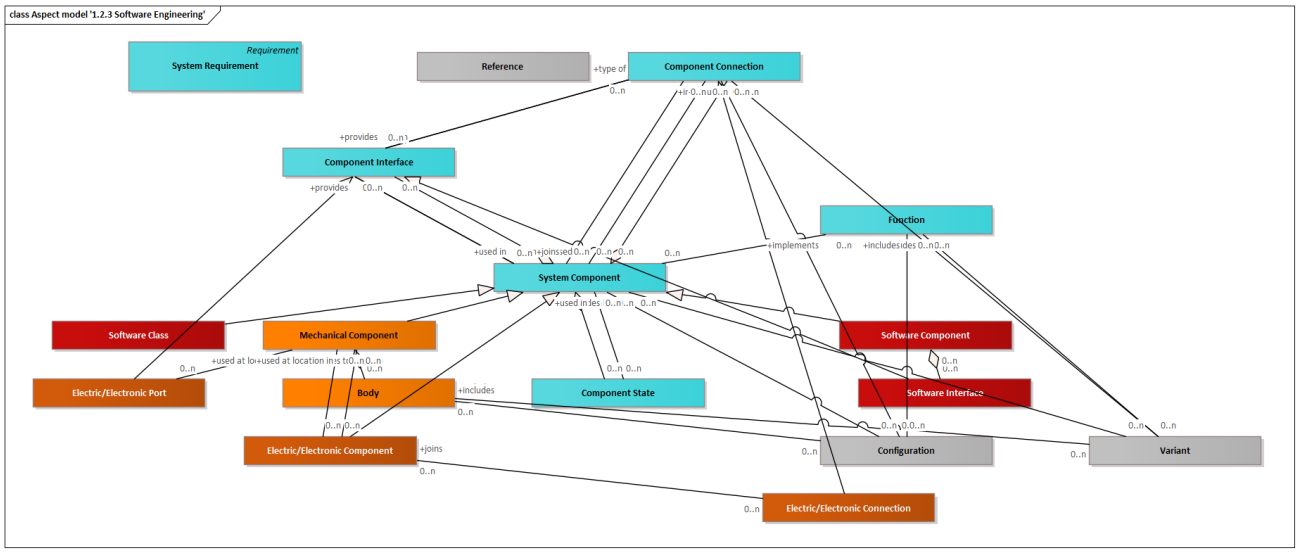


Figure 206 Aspect model '1.2.3 Software Engineering'

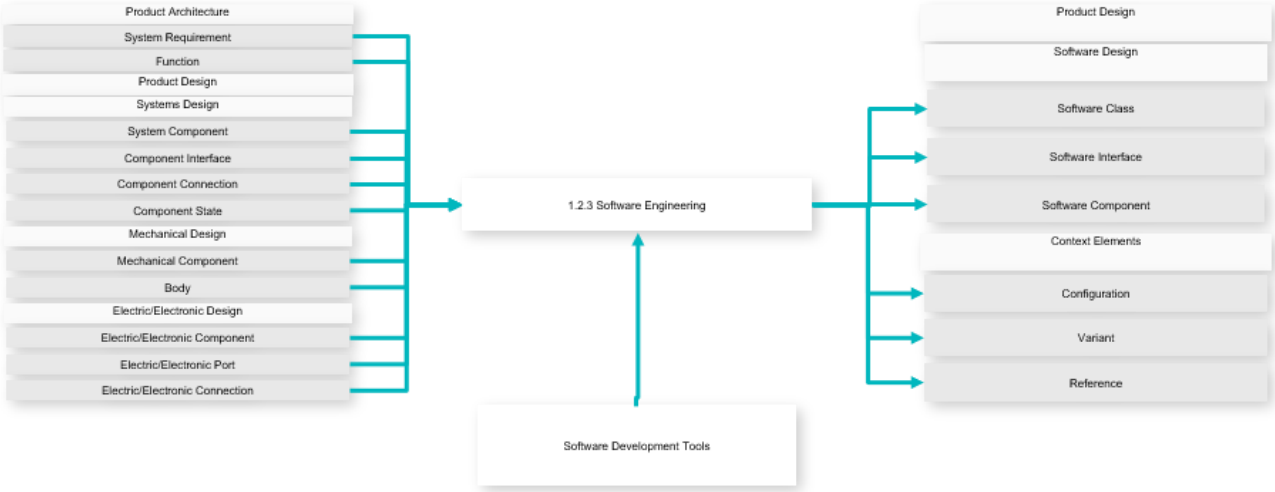


Figure 207 Use-case Diagram

Inputs	Function Component Interface System Requirement System Component Mechanical Component Electric/Electronic Component Component Connection Component State Electric/Electronic Port Electric/Electronic Connection Body
Outputs	Configuration Variant Software Class Software Component Software Interface Reference

8.5.5.2.1.3.2.5. Electrical/Mechanical Design Collaboration

In Electrical/Mechanical Design Collaboration, both design disciplines involved will collaborate in a common 3D space in order to adjust and coordinate allocation, dimensioning and positioning of components.

Figure 208 Required Input

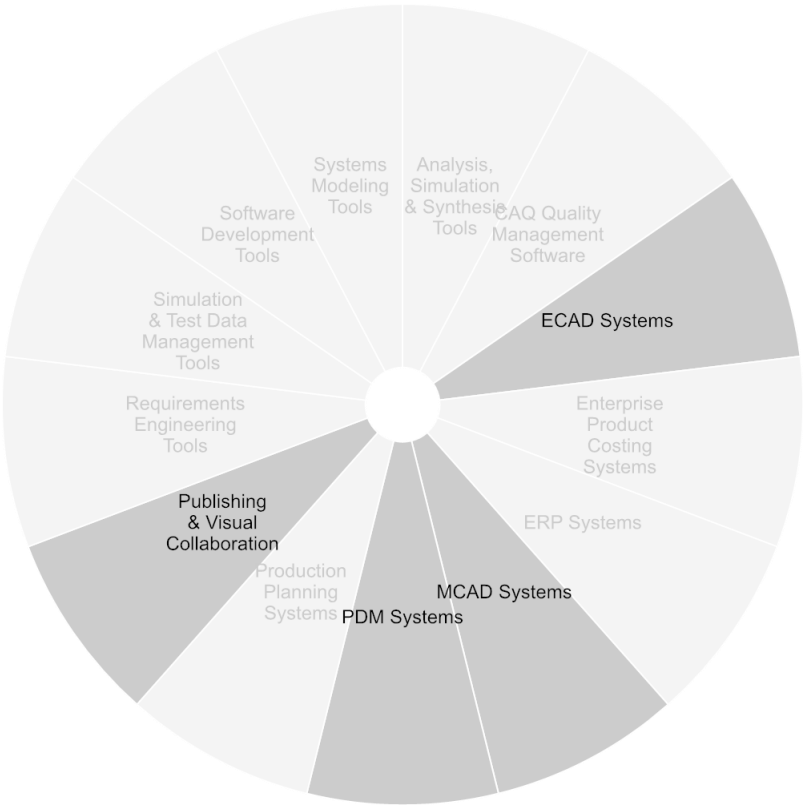


Figure 209 Used Tools

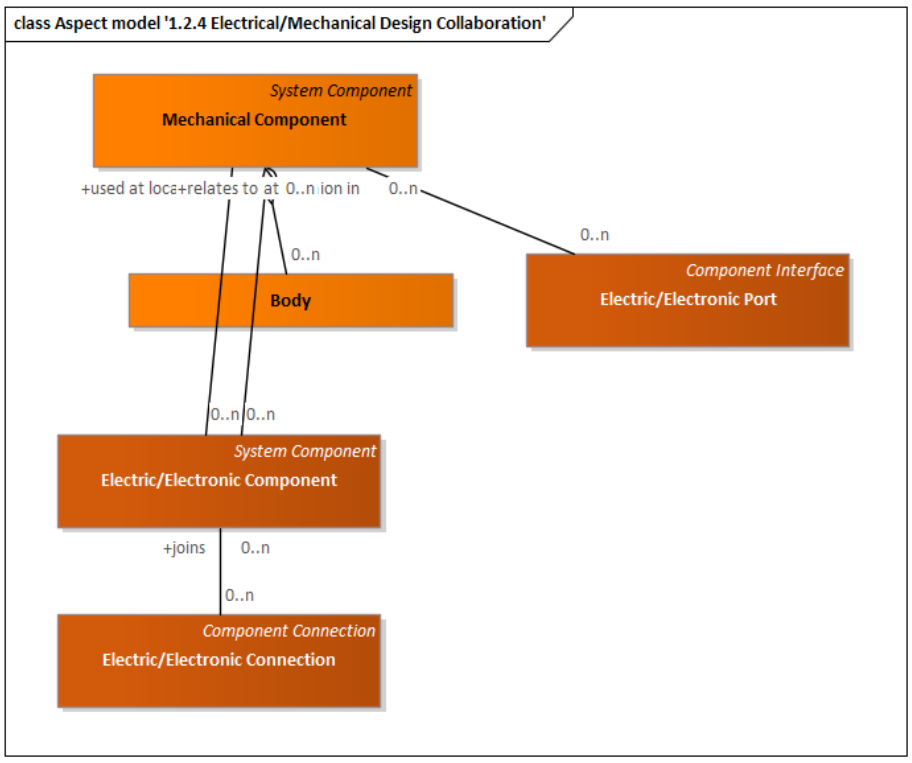


Figure 210 Aspect model '1.2.4 Electrical Mechanical Design Collaboration'

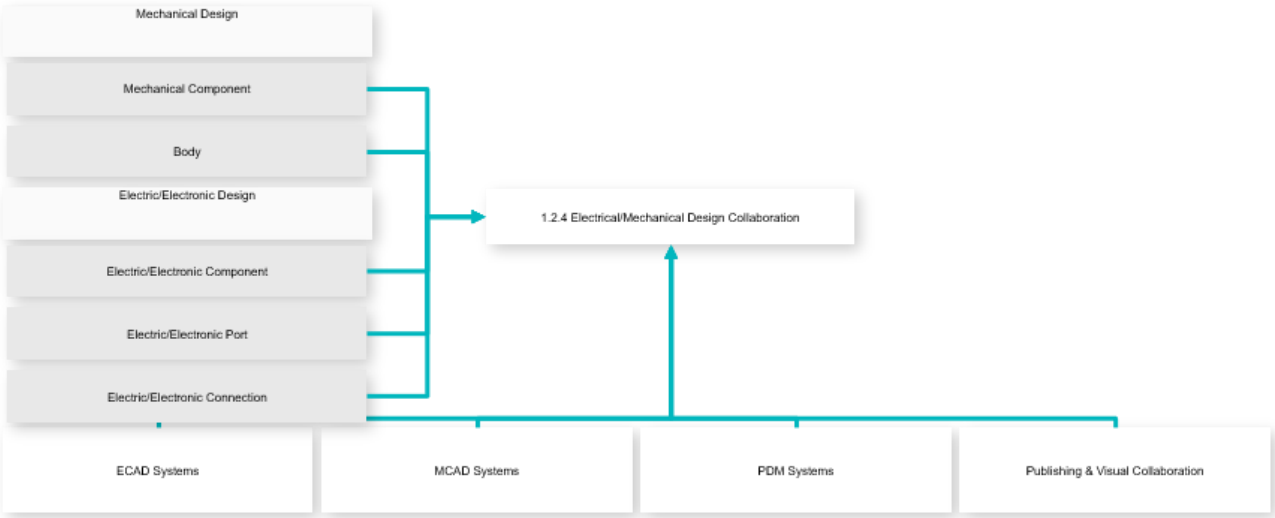


Figure 211 Use-case Diagram

Inputs	Mechanical Component Electric/Electronic Component Electric/Electronic Port Electric/Electronic Connection Body

8.5.5.2.1.3.3. Verification & Validation using Simulation

Modeling and simulation (M&S) is the domain-specific use of models (e.g., physical, mathematical, or logical representation of a system, entity, phenomenon, or process) as a basis for simulations to develop data utilized for management or technical decision making.

Simulation may include finite elements analysis, fluid analysis, DMU simulation, kinematics simulation, as well as simulation based on software prototypes. It also includes X-in-the-Loop (XiL), using integrated design and simulation models and capabilities.

Verification aims at the demonstration of product compliance to the specified design requirements.

Validation aims at the demonstration of conformance to stakeholder requirements.

Description based on: https://en.wikipedia.org/wiki/Modeling_and_simulation, ISO 29148, ISO 15288

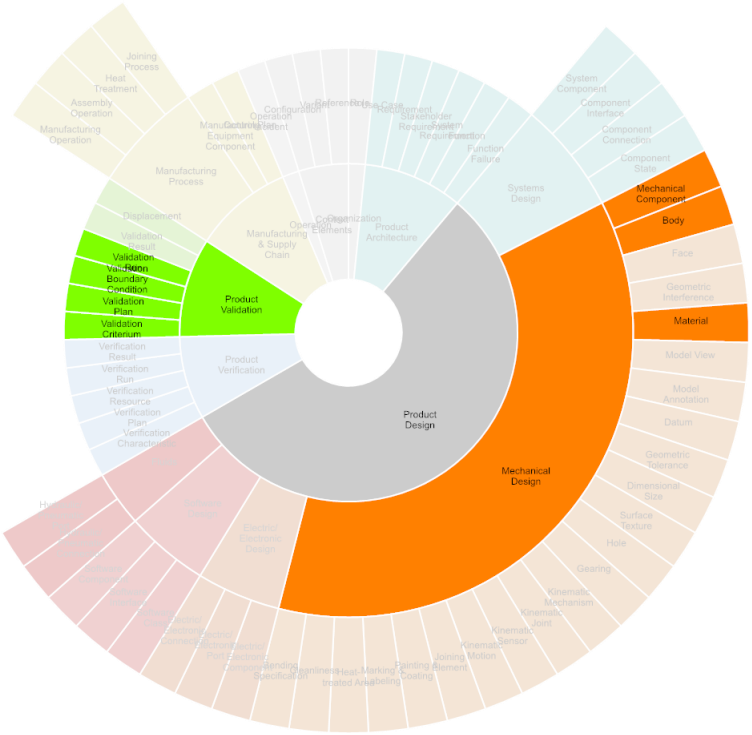


Figure 212 Required Input

8.5.5.2.1.3.3.1. Plan and prepare Simulation

Planning and preparation of a simulation includes the specification of the simulation objectives based on product requirements and functions. This is followed by the definition of the simulation boundary conditions and the createion of simulation models according to simulation objectives.

Figure 213 Required Input

Figure 214 Used Tools

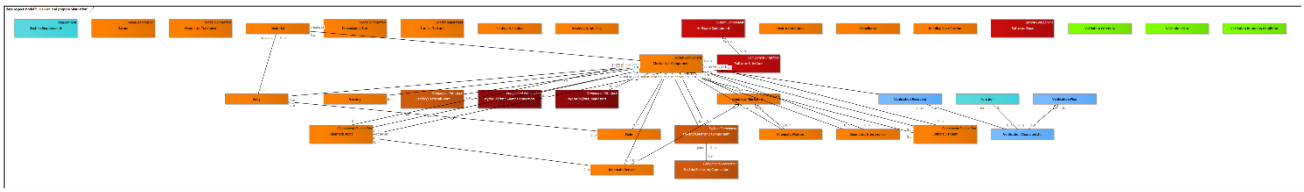


Figure 215 Aspect model '1.3.1 Plan and prepare Simulation'

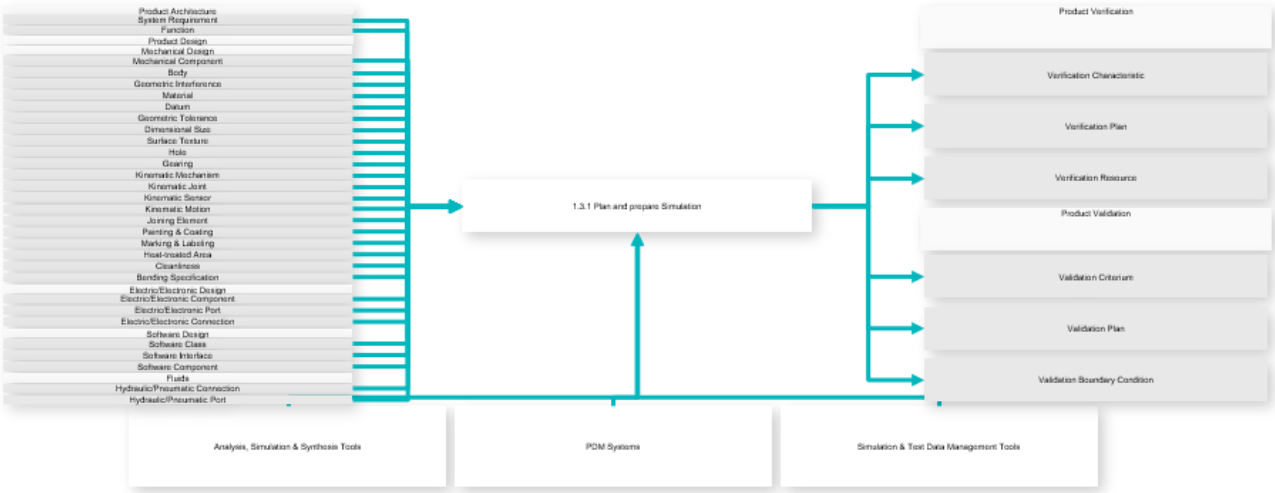


Figure 216 Use-case Diagram

Inputs	Function System Requirement Mechanical Component Electric/Electronic Component Software Class Software Component Electric/Electronic Port Software Interface Hydraulic/Pneumatic Port Kinematic Joint Joining Element Electric/Electronic Connection Hydraulic/Pneumatic Connection Material Body Geometric Interference Hole Gearing Kinematic Mechanism Kinematic Sensor Kinematic Motion Datum Geometric Tolerance Dimensional Size Surface Texture Painting & Coating Marking & Labeling Heat-treated Area Cleanliness Bending Specification

Outputs	Validation Criterium Verification Characteristic Verification Resource Verification Plan Validation Plan Validation Boundary Condition

8.5.5.2.1.3.3.2. Perform and document Simulation

Simulation execution includes performing the simulation as well as documenting and summarizing the simulation results.

Figure 217 Required Input

Figure 218 Used Tools

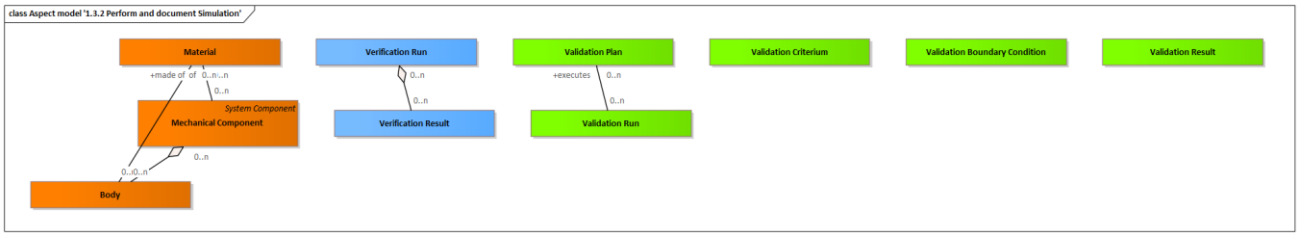


Figure 219 Aspect model '1.3.2 Perform and document Simulation'

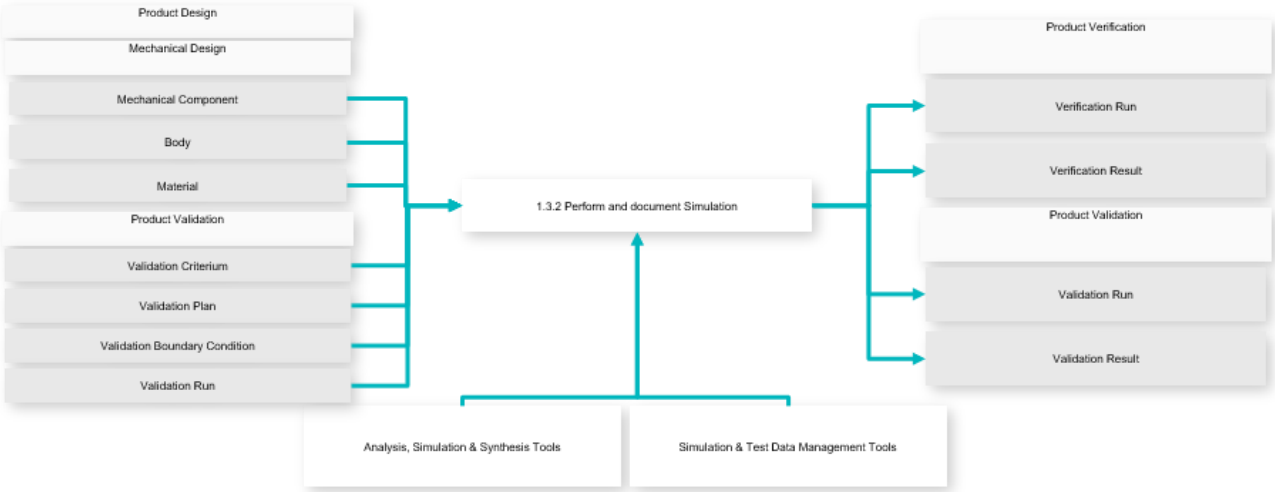


Figure 220 Use-case Diagram

Inputs	Validation Criterium Mechanical Component Material Body Validation Plan Validation Boundary Condition Validation Run
Outputs	Verification Run Verification Result Validation Run Validation Result

8.5.5.2.1.3.4. Verification & Validation using Test
While simulation is based on virtual models, testing is based on prototypes and samples.
Verification is to demonstrate product compliance to the specified design requirements. Validation is to demonstrate conformance to stakeholder requirements.
Description based on ISO 29148, resp. ISO 15288

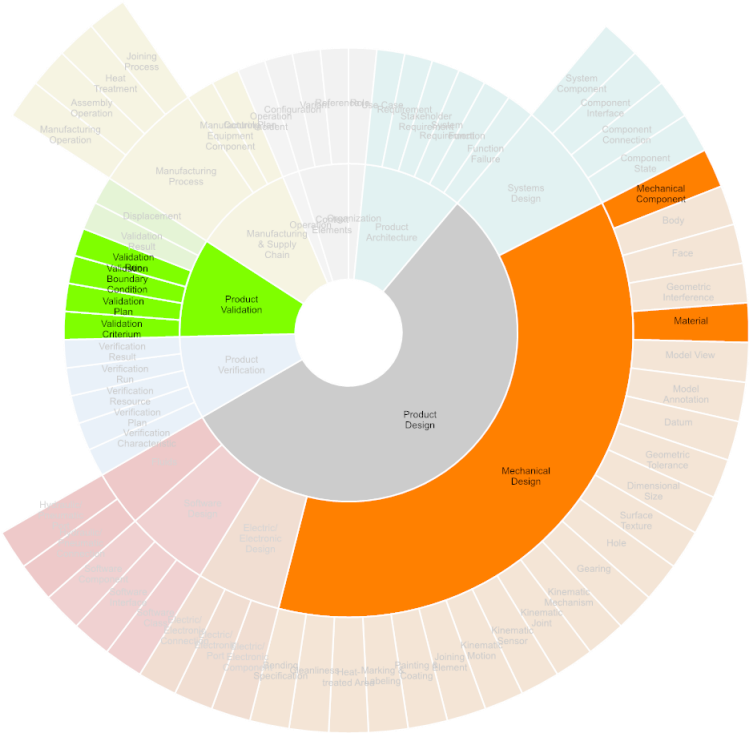


Figure 221 Required Input

8.5.5.2.1.3.4.1. Prepare Testing

Testing preparation includes the definition of product requirements and functions to be validated, the specification, design and manufacturing of samples, prototypes and test rigs.

Figure 222 Required Input

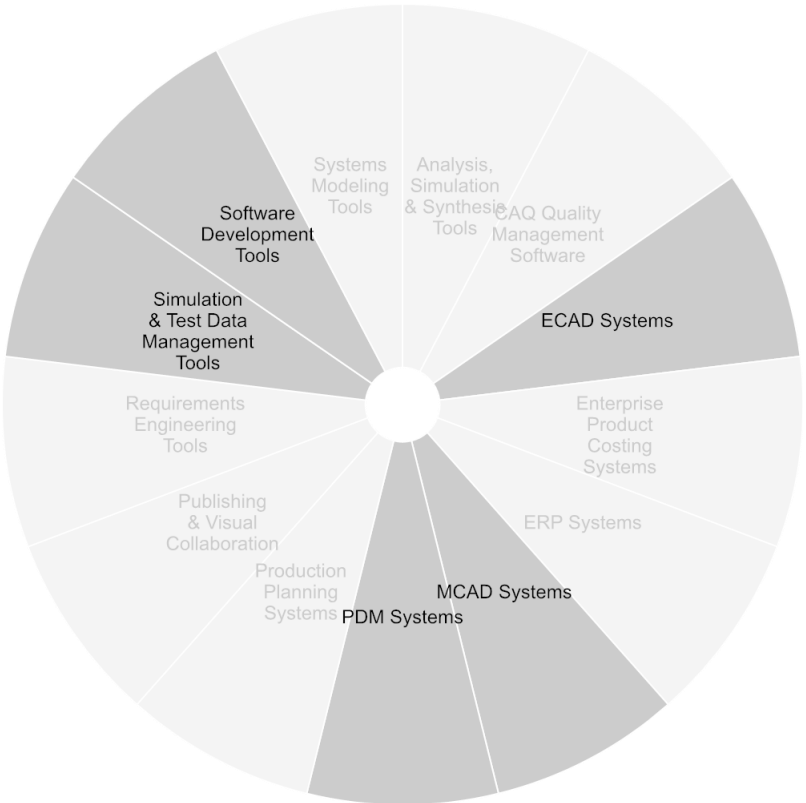


Figure 223 Used Tools

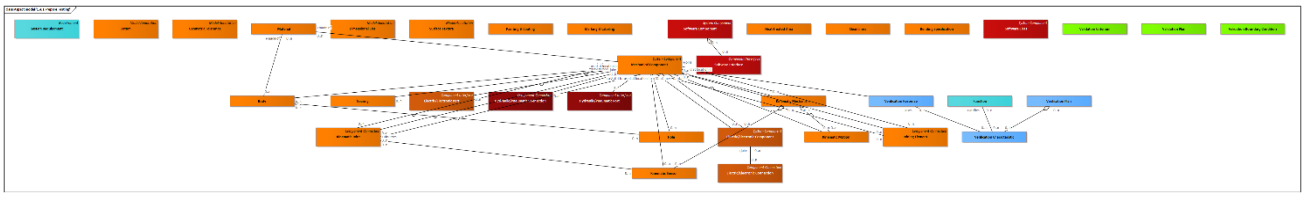


Figure 224 Aspect model '1.4.1 Prepare Testing'

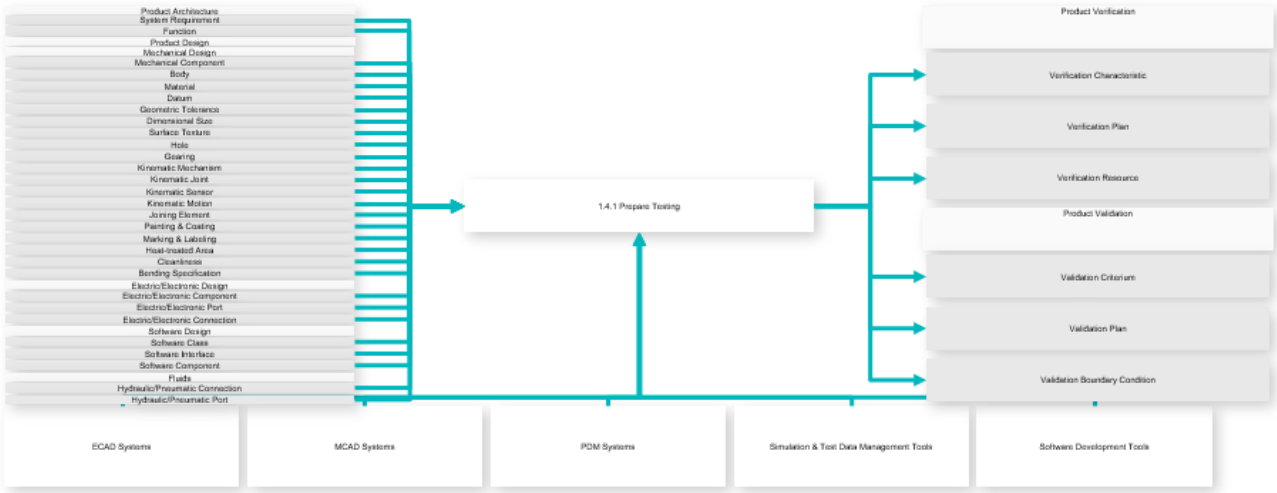


Figure 225 Use-case Diagram

Inputs	Function System Requirement Mechanical Component Electric/Electronic Component Software Class Software Component Electric/Electronic Port Software Interface Hydraulic/Pneumatic Port Kinematic Joint Joining Element Electric/Electronic Connection Hydraulic/Pneumatic Connection Material Body Hole Gearing Kinematic Mechanism Kinematic Sensor Kinematic Motion Datum Geometric Tolerance Dimensional Size Surface Texture Painting & Coating Marking & Labeling Heat-treated Area Cleanliness Bending Specification

Outputs	Validation Criterium Verification Characteristic Verification Resource Verification Plan Validation Plan Validation Boundary Condition

8.5.5.2.1.3.4.2. Perform and document Testing

Testing execution includes scheduling tests, allocating the test rig, executing and documenting the test.

Figure 226 Required Input

Figure 227 Used Tools

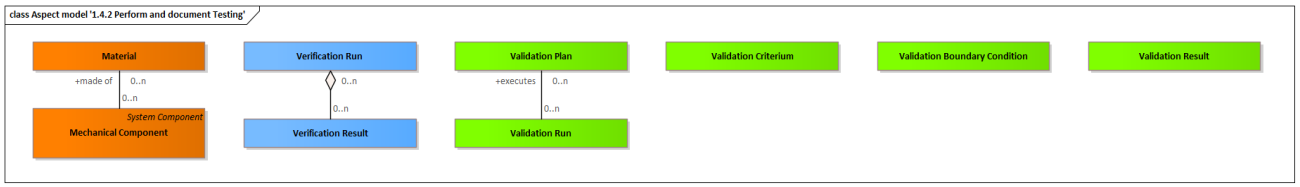


Figure 228 Aspect model '1.4.2 Perform and document Testing'

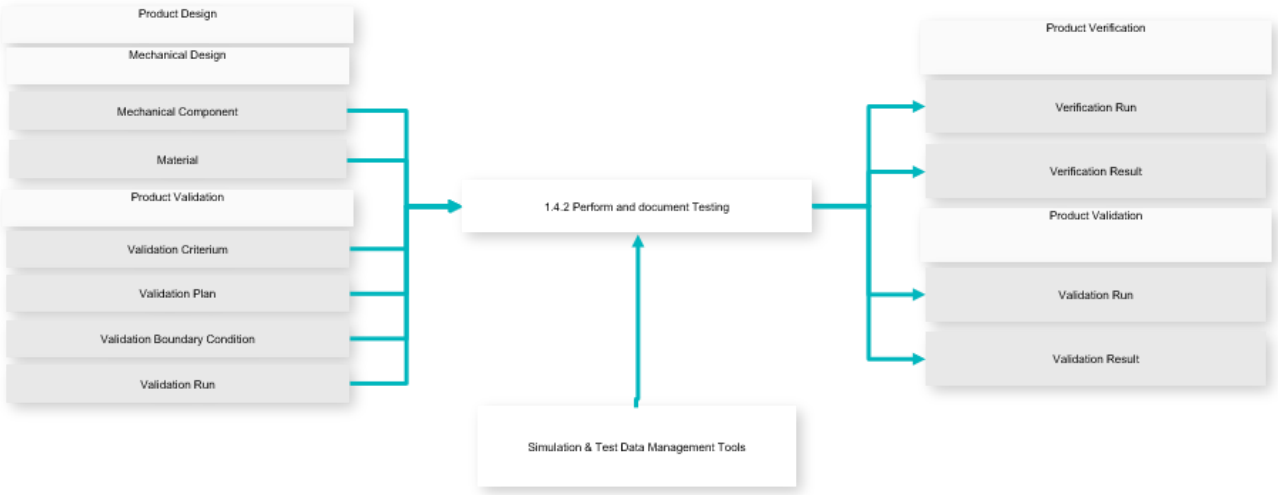


Figure 229 Use-case Diagram

Inputs	Validation Criterium Mechanical Component

	Material Validation Plan Validation Boundary Condition Validation Run
Outputs	Verification Run Verification Result Validation Run Validation Result

8.5.5.2.1.4. Production Development

The manufacturing or production engineer's primary focus is to turn raw material into an updated or new product in the most effective, efficient & economic way possible.
Manufacturing Engineering is based on core industrial engineering and mechanical engineering skills, adding important elements from mechatronics, commerce, economics and business management.
Source: https://en.wikipedia.org/wiki/Manufacturing_engineering

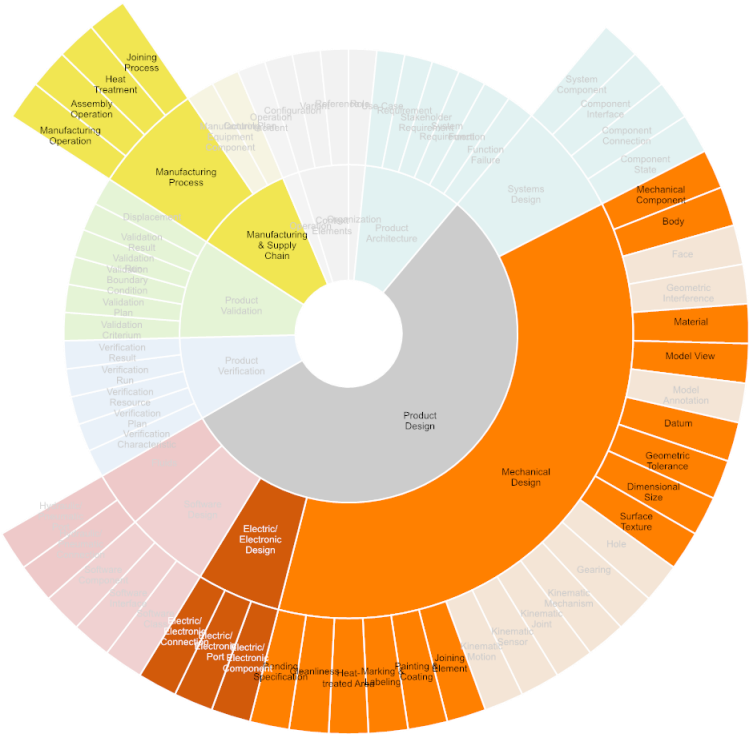


Figure 230 Required Input

Figure 231 Used Tools

8.5.5.2.1.4.1. Process Planning

The task in the manufacturing process planning domain is to find a plan to manufacture a set of parts.
Source: https://www.cs.cmu.edu/afs/cs/project/jair/pub/volume15/ambite01a-html/node23.html

Figure 232 Required Input

Figure 233 Used Tools

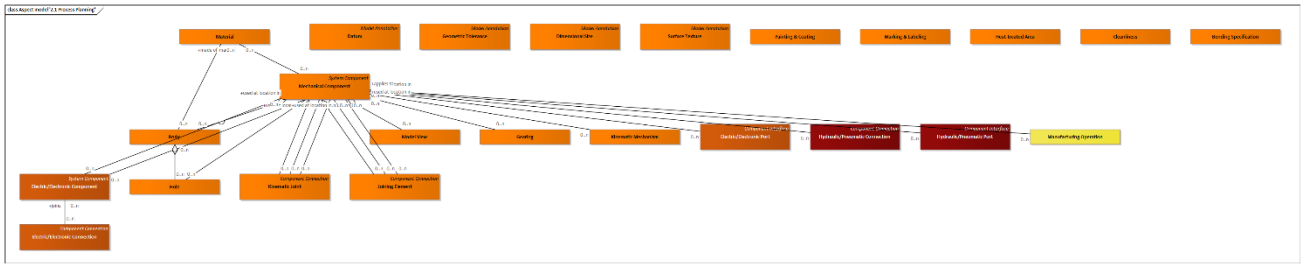


Figure 234 Aspect model '2.1 Process Planning'

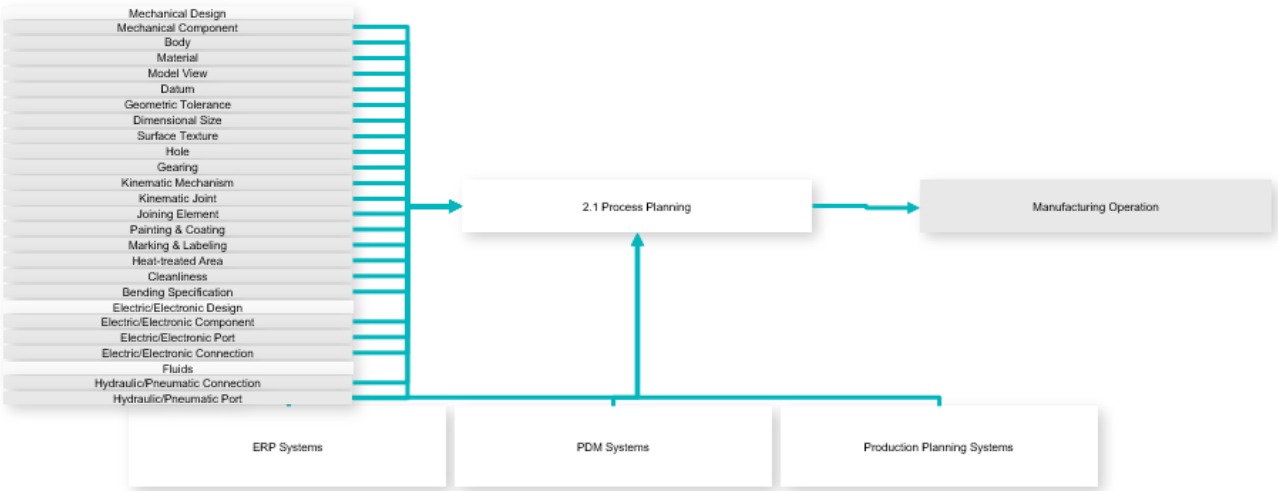


Figure 235 Use-case Diagram

Inputs	Mechanical Component Electric/Electronic Component Electric/Electronic Port Hydraulic/Pneumatic Port Kinematic Joint Joining Element Electric/Electronic Connection Hydraulic/Pneumatic Connection Material Body Model View Hole Gearing Kinematic Mechanism Datum Geometric Tolerance Dimensional Size Surface Texture Painting & Coating Marking & Labeling Heat-treated Area Cleanliness Bending Specification
Outputs	Manufacturing Operation

8.5.5.2.1.4.2. Inspection Planning

Includes the selection of relevant inspection criteria from product and production specifications, identification and assignment of appropriate inspection methods and equipment, as well as the allocation of appropriate inspection criteria per inspection event (first-article inspection, re-qualification, inspection in production, etc.)

Figure 236 Required Input

Figure 237 Used Tools

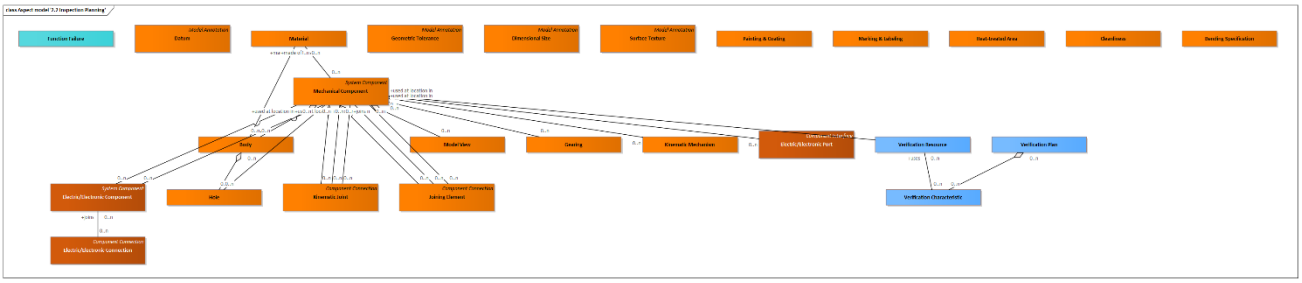


Figure 238 Aspect model '2.2 Inspection Planning'

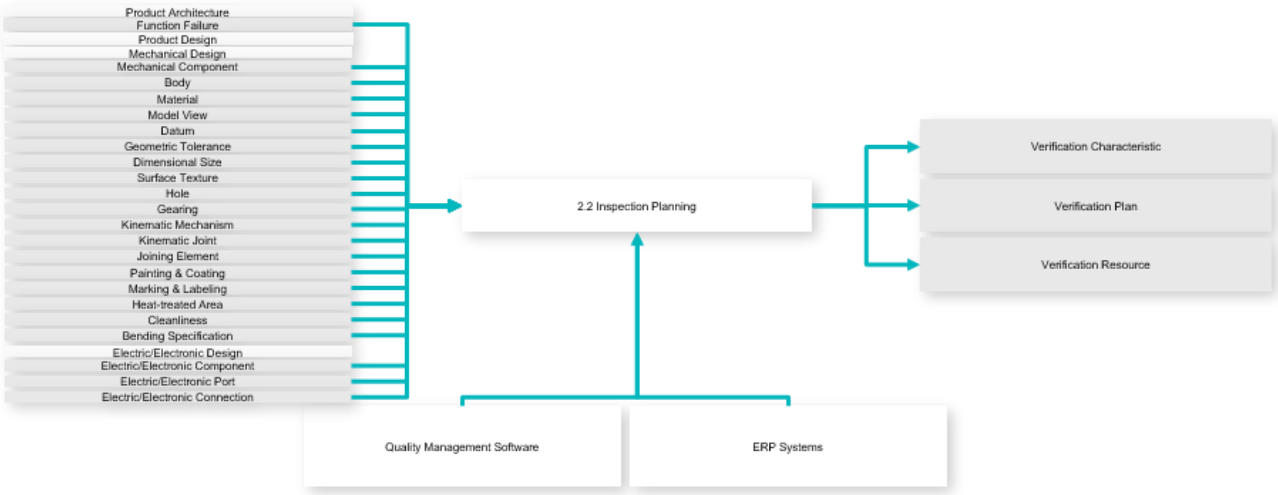


Figure 239 Use-case Diagram

Inputs	Function Failure Mechanical Component Electric/Electronic Component Electric/Electronic Port Kinematic Joint Joining Element Electric/Electronic Connection Material Body Model View Hole Gearing Kinematic Mechanism Datum Geometric Tolerance Dimensional Size Surface Texture Painting & Coating Marking & Labeling Heat-treated Area Cleanliness Bending Specification
Outputs	Verification Characteristic Verification Resource Verification Plan

8.5.5.2.1.4.3. Cost Calculation

Includes the calculation of product cost using relevant information from product and production specifications, including dimensions, material, mass, geometric & topological complexity, surface finish including paint or coatings.

Figure 240 Required Input

Figure 241 Used Tools

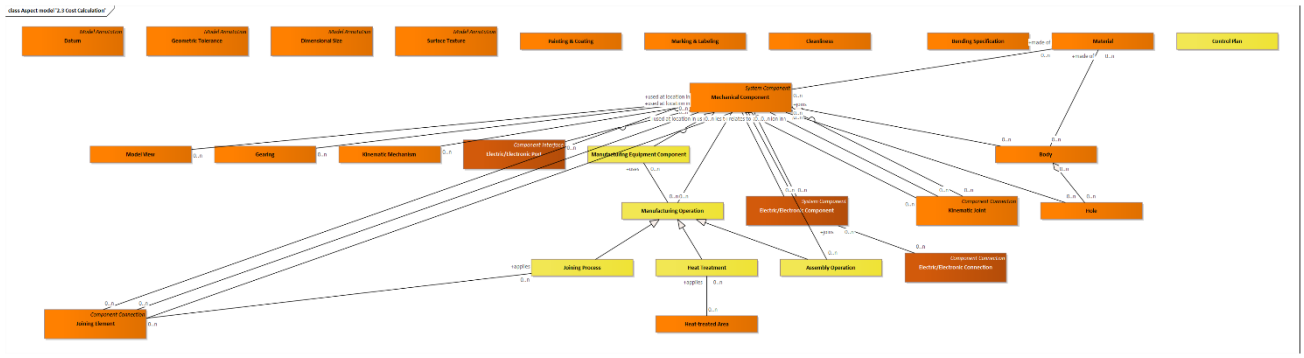


Figure 242 Aspect model '2.3 Cost Calculation'

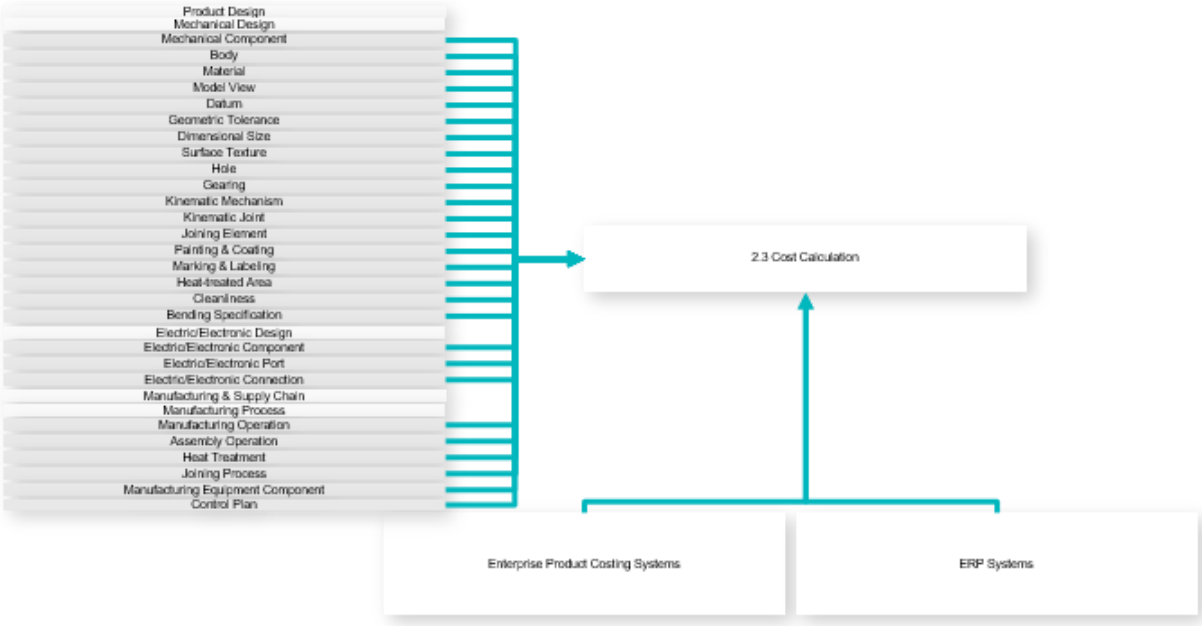


Figure 243 Use-case Diagram

Inputs	Manufacturing Operation Mechanical Component Electric/Electronic Component Electric/Electronic Port Kinematic Joint Joining Element Electric/Electronic Connection Manufacturing Equipment Component Material Body Model View Hole Gearing Kinematic Mechanism Assembly Operation Datum Geometric Tolerance Dimensional Size Surface Texture Joining Process Painting & Coating Marking & Labeling Heat-treated Area Heat Treatment Cleanliness Bending Specification Control Plan

8.5.5.2.1.4.4. Plant Layout

Defines the logical and physical layout of a production line in a plant or buiding using information from product and production specifications, including production steps, material handling, as well as the physical dimensions of product components, tools, fixtures and equipment within the topology and systems of the manufacturing site or building.

Figure 244 Required Input

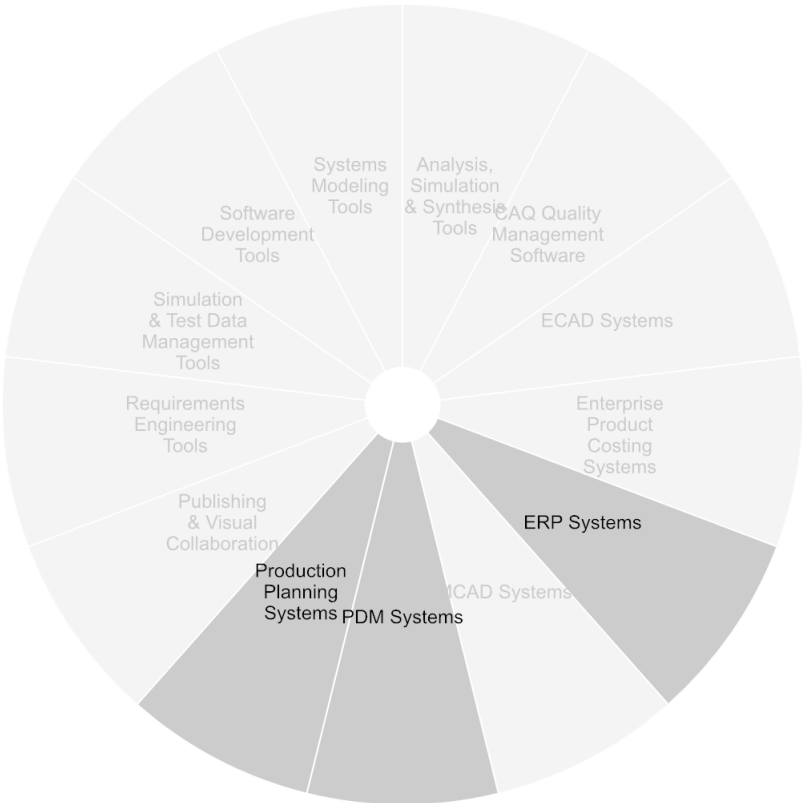


Figure 245 Used Tools

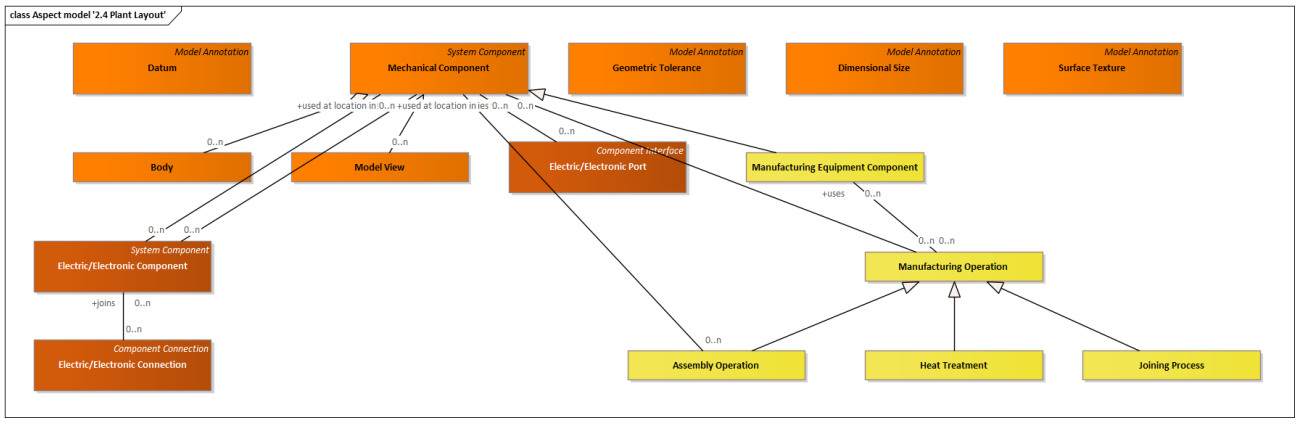


Figure 246 Aspect model '2.4 Plant Layout'

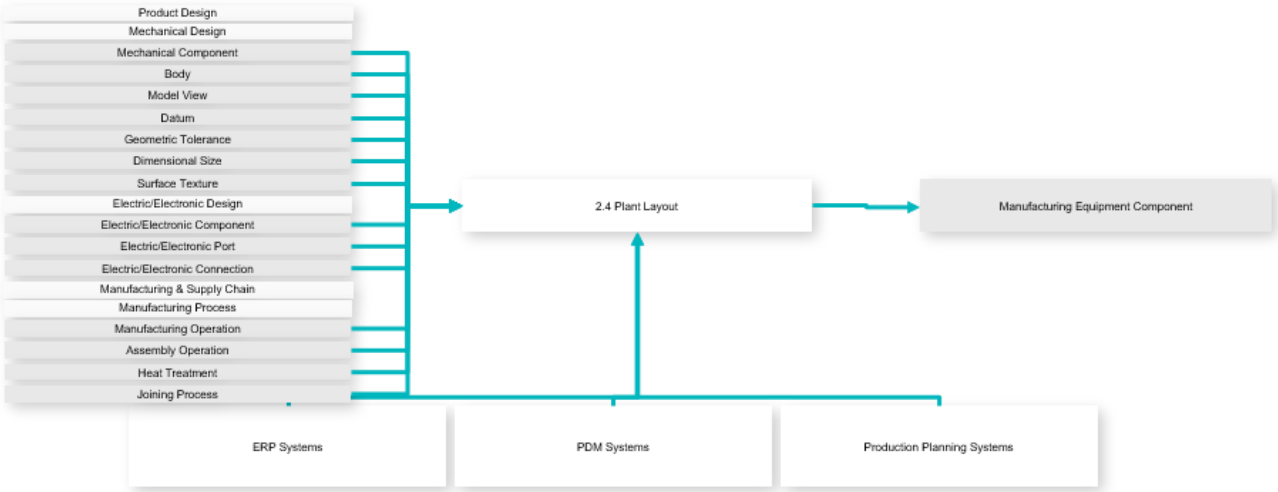


Figure 247 Use-case Diagram

Inputs	Manufacturing Operation Mechanical Component Electric/Electronic Component Electric/Electronic Port Electric/Electronic Connection Body Model View Assembly Operation Datum Geometric Tolerance Dimensional Size Surface Texture Joining Process Heat Treatment
Outputs	Manufacturing Equipment Component

8.5.5.2.1.4.5. Tools & Equipment Design

The design of tools and equipment aligned with product design, the manufacturing process plan and plant layout.

Figure 248 Required Input

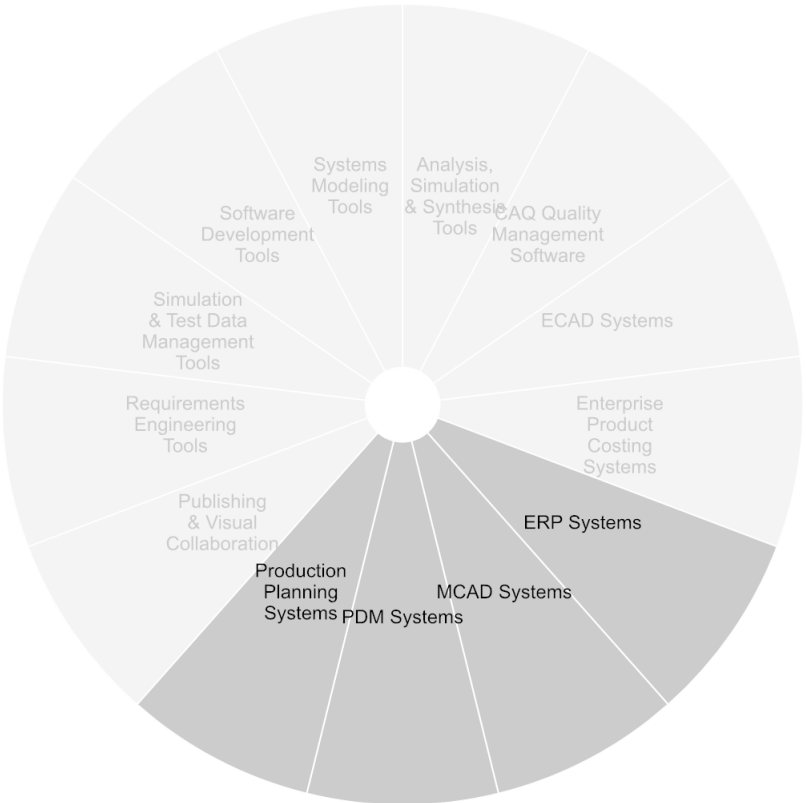


Figure 249 Used Tools

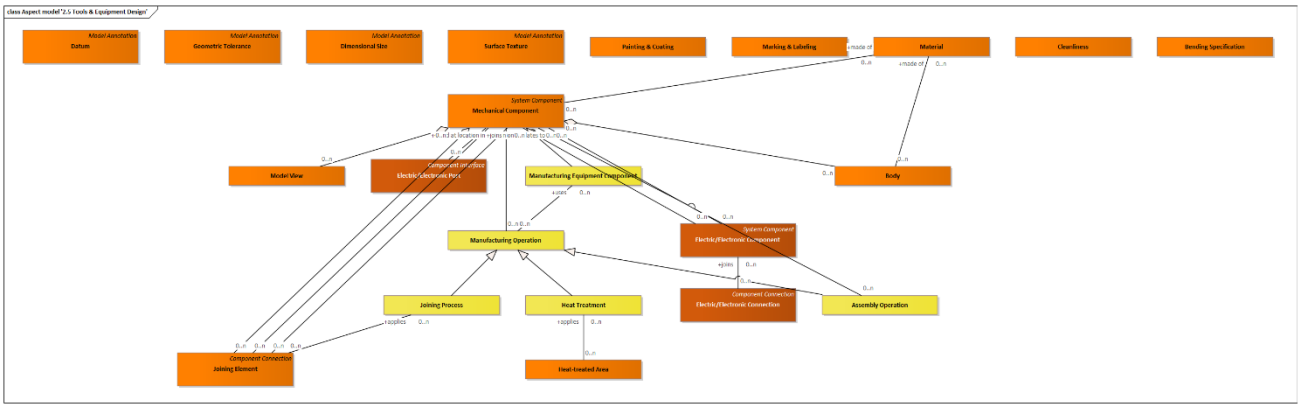


Figure 250 Aspect model '2.5 Tools & Equipment Design'

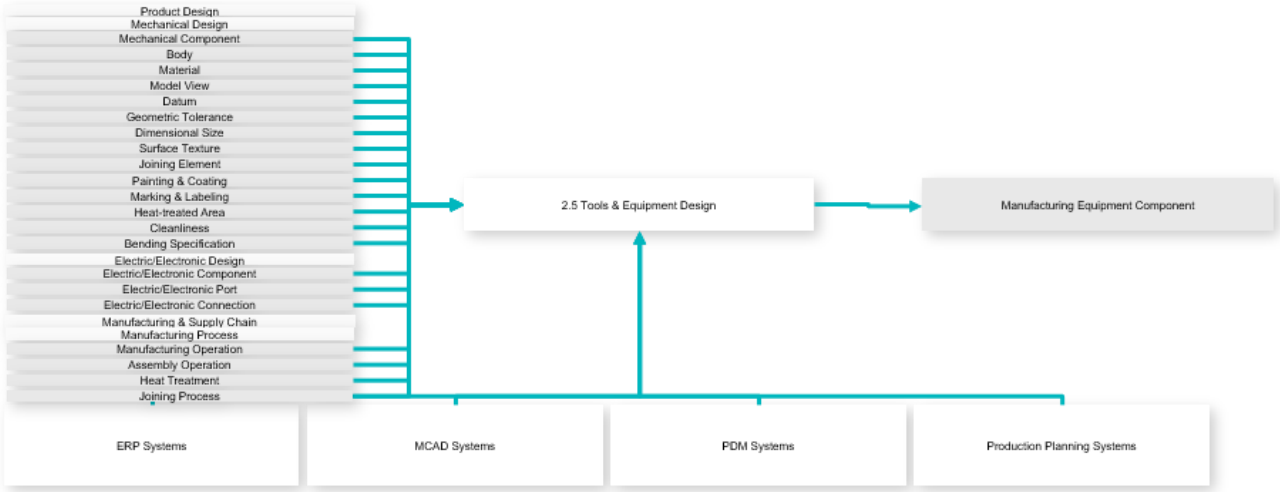


Figure 251 Use-case Diagram

Inputs	Manufacturing Operation Mechanical Component Electric/Electronic Component Electric/Electronic Port Joining Element Electric/Electronic Connection Material Body Model View Assembly Operation Datum Geometric Tolerance Dimensional Size Surface Texture Joining Process Painting & Coating Marking & Labeling Heat-treated Area Heat Treatment Cleanliness Bending Specification
Outputs	Manufacturing Equipment Component

8.5.5.2.1.5. Manufacturing & Supply Chain

see: Manufacturing & Supply Chain

8.5.5.2.1.5.1. Parts Manufacturing

Manufacturing of single parts using a variety of production processes and methods, each with specific sets of data, such as CNC programs, bending specification, holes specifications, joining features, etc.

Figure 252 Required Input

Figure 253 Used Tools

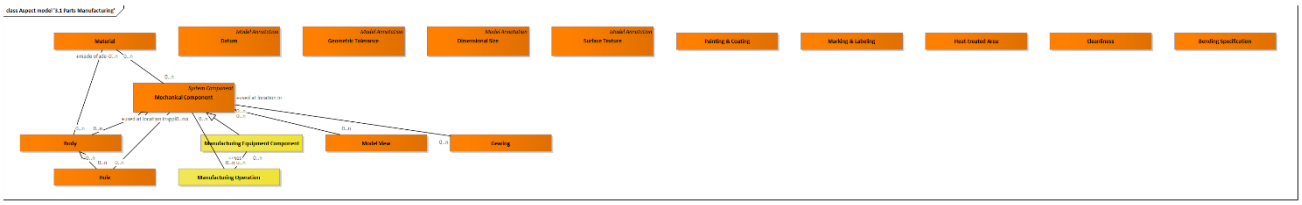


Figure 254 Aspect model '3.1 Parts Manufacturing'

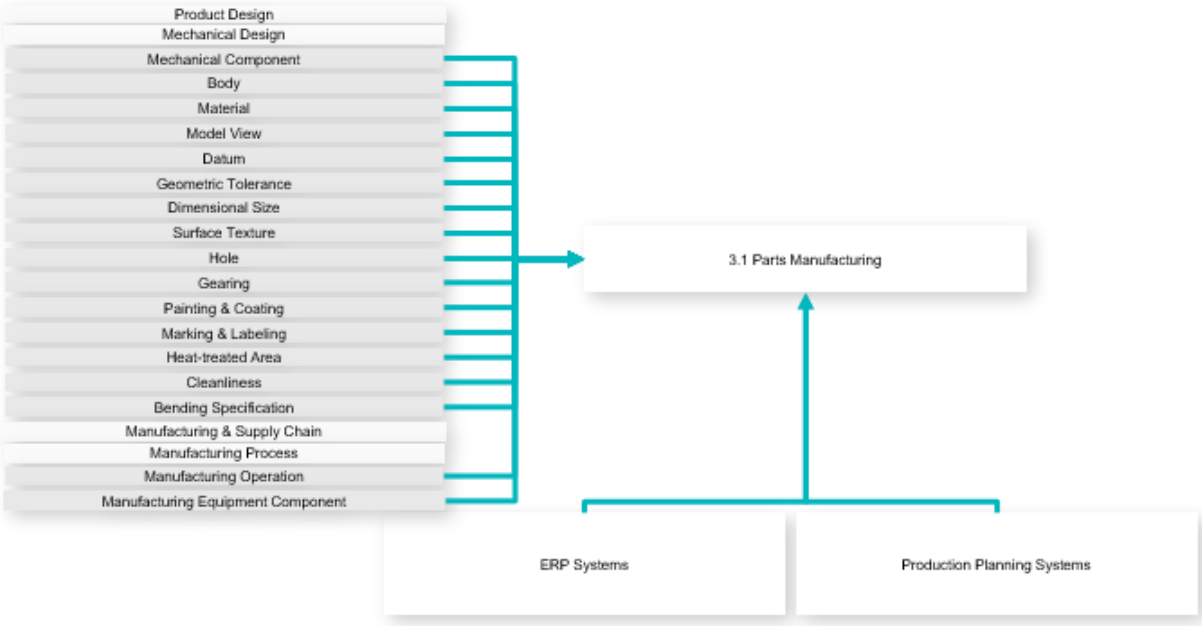


Figure 255 Use-case Diagram

Inputs	Manufacturing Operation Mechanical Component Manufacturing Equipment Component Material Body Model View Hole Gearing Datum Geometric Tolerance Dimensional Size Surface Texture Painting & Coating Marking & Labeling Heat-treated Area Cleanliness Bending Specification

8.5.5.2.1.5.2. Product Assembly

Assembly of components or products from single parts or sub-components according to a production plan, using all kinds of joinig and bonding.

Figure 256 Required Input

Figure 257 Used Tools

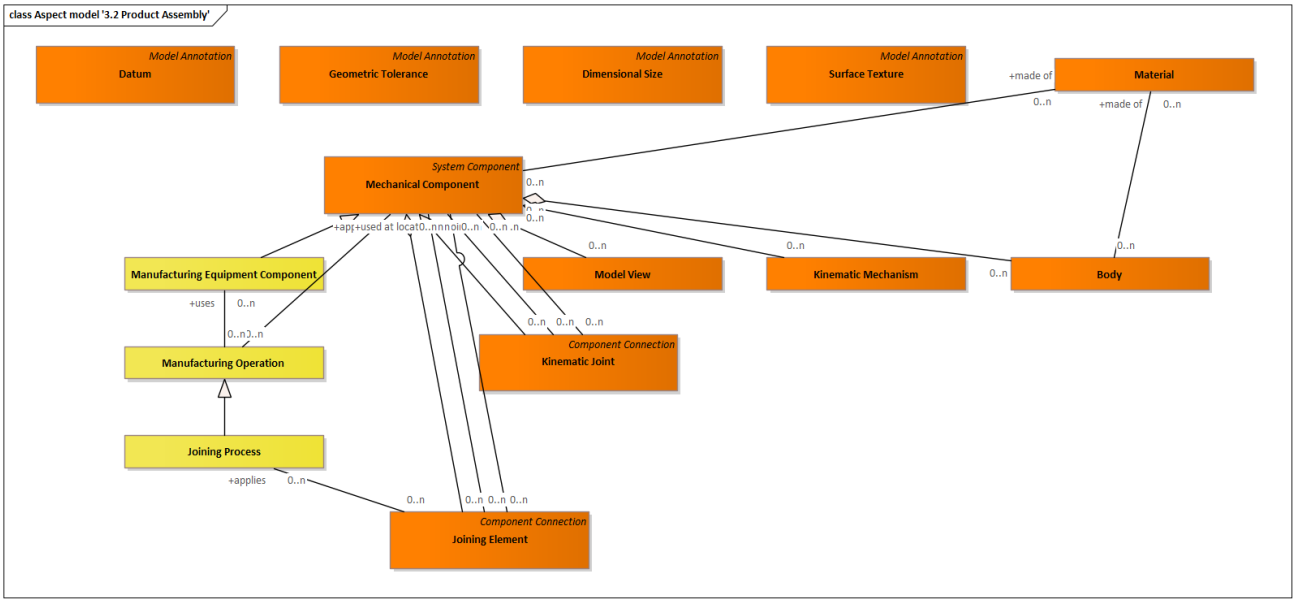


Figure 258 Aspect model '3.2 Product Assembly'

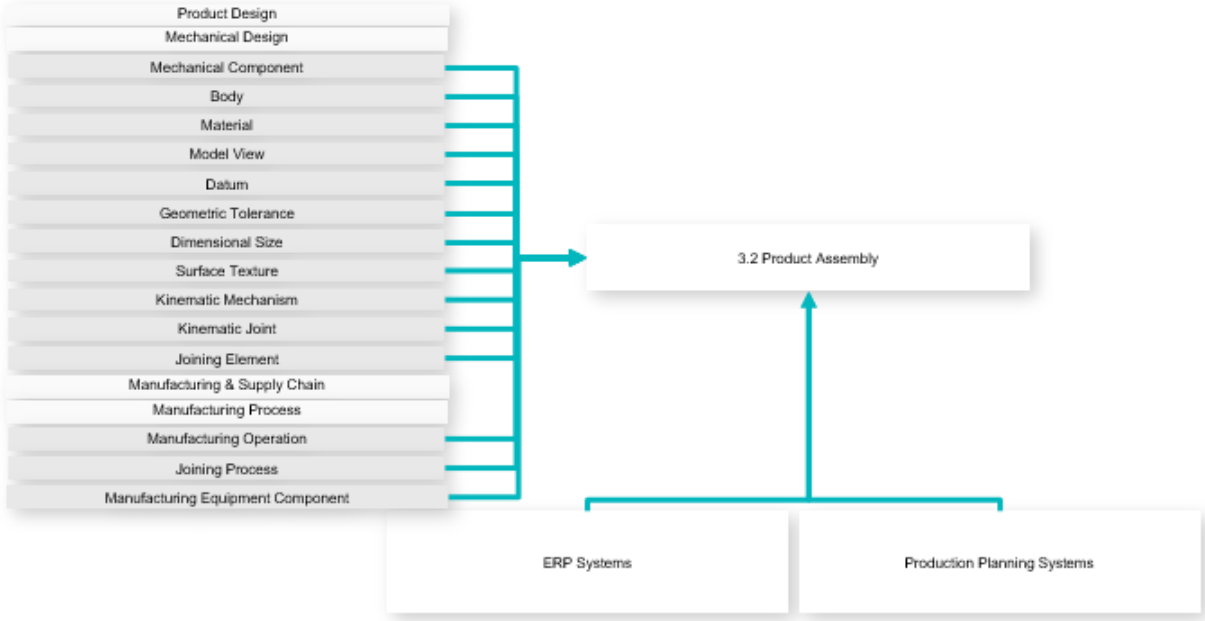


Figure 259 Use-case Diagram

Inputs	Manufacturing Operation Mechanical Component Kinematic Joint Joining Element Manufacturing Equipment Component Material Body Model View Kinematic Mechanism Datum Geometric Tolerance

	Dimensional Size Surface Texture Joining Process

8.5.5.2.1.5.3. Quality Inspection

Includes the inspection of single parts, components or products during or after the manufacturing process, as well as the documentation of inspection results.

Figure 260 Required Input

Figure 261 Used Tools

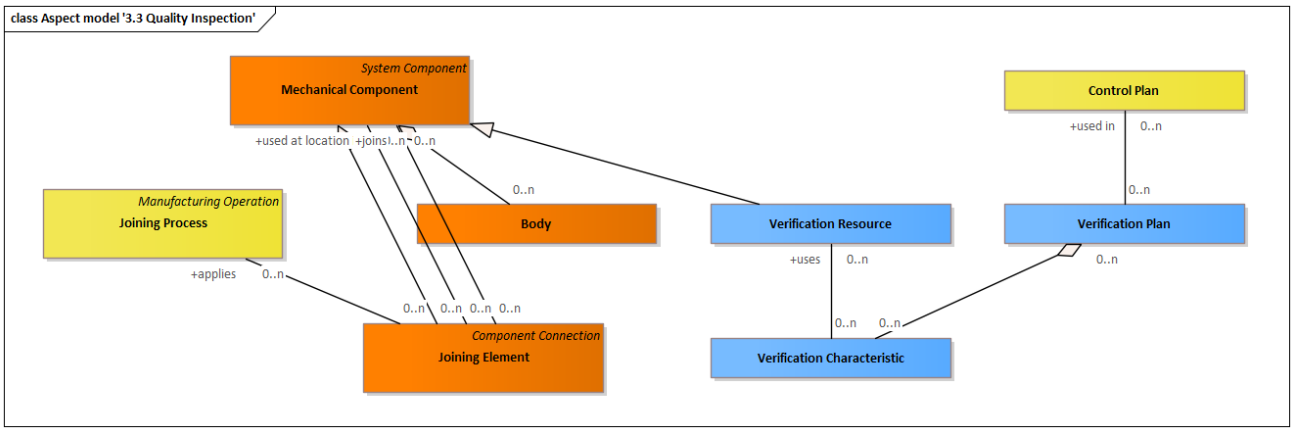


Figure 262 Aspect model '3.3 Quality Inspection'

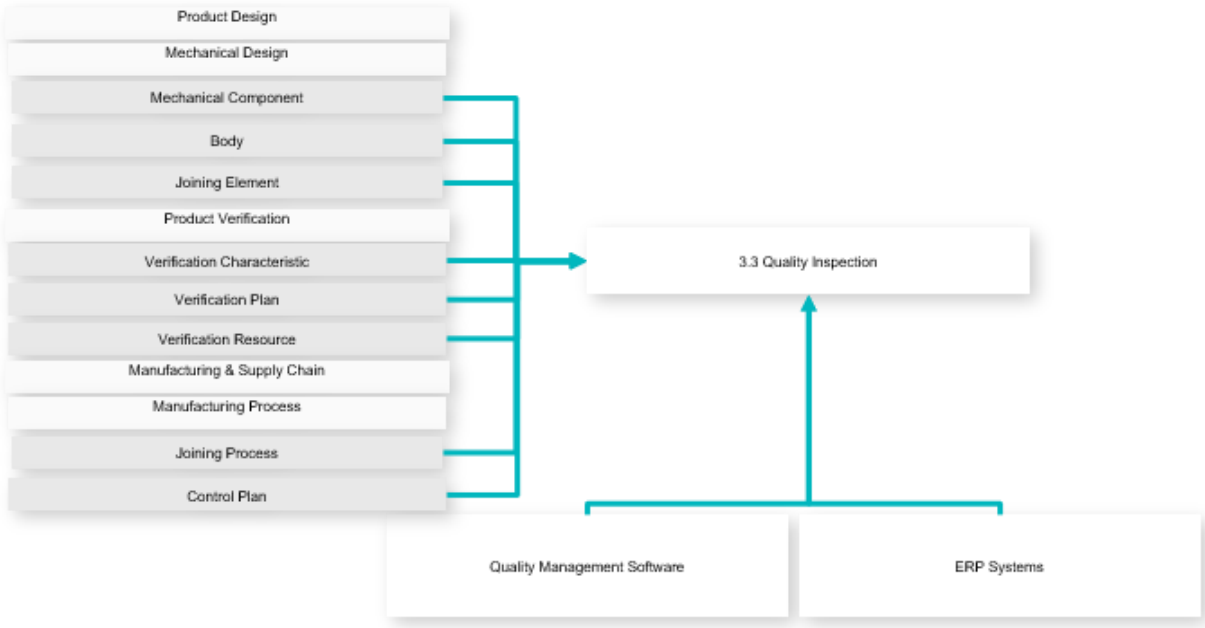


Figure 263 Use-case Diagram

Inputs	Verification Characteristic Mechanical Component Joining Element Verification Resource Body Joining Process Control Plan Verification Plan

8.5.5.2.1.5.4. Packaging & Logistics

Includes the definition of all kinds of packaging and transport equipment to ensure a product or its components will be supplied to their destination safely.

Figure 264 Required Input

Figure 265 Used Tools

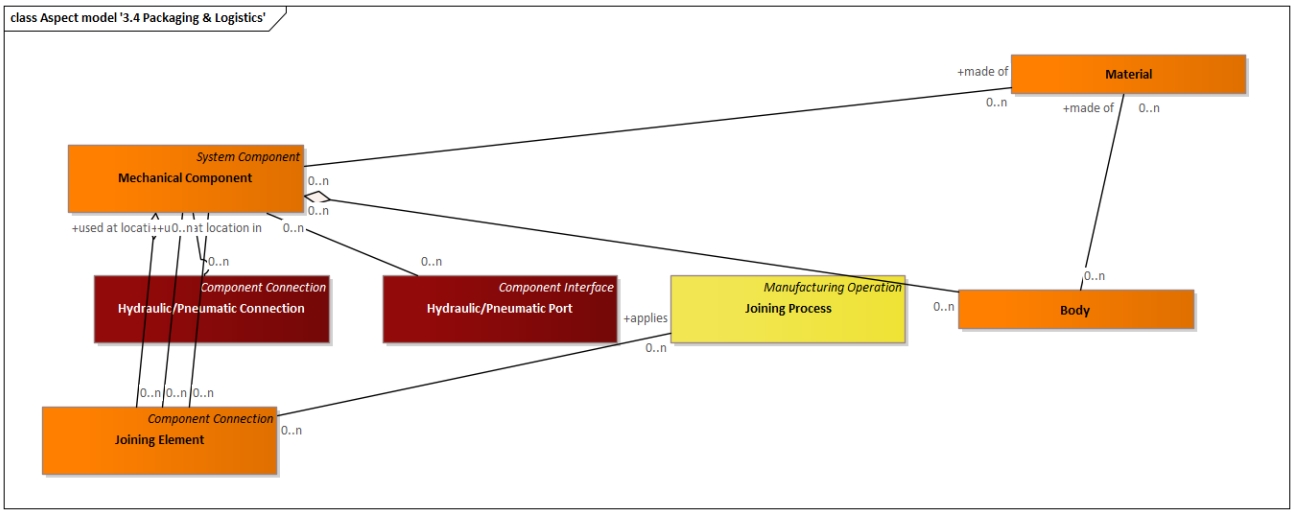


Figure 266 Aspect model '3.4 Packaging & Logistics'

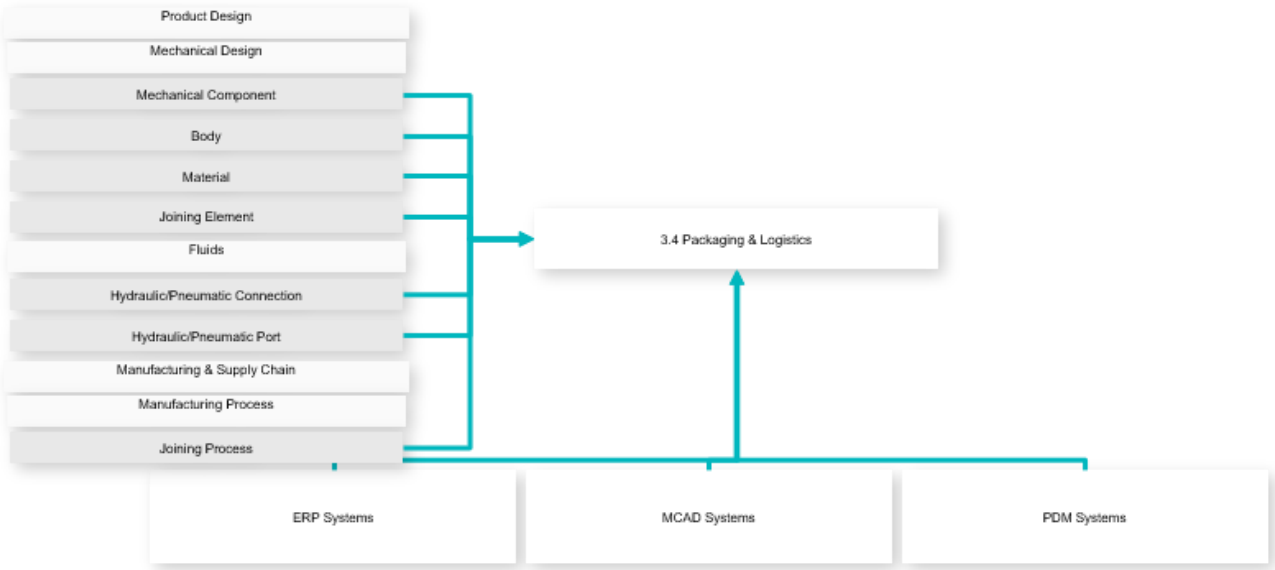


Figure 267 Use-case Diagram

Inputs	Mechanical Component Hydraulic/Pneumatic Port Joining Element Hydraulic/Pneumatic Connection Material Body Joining Process

8.5.5.2.1.6. Product Operation

Includes all phases and activities in the operation phase of a product.

Figure 268 Required Input

8.5.5.2.1.6.1. Product Usage

Usage of a product according to the designated purpose and function.

Figure 269 Required Input

Figure 270 Used Tools

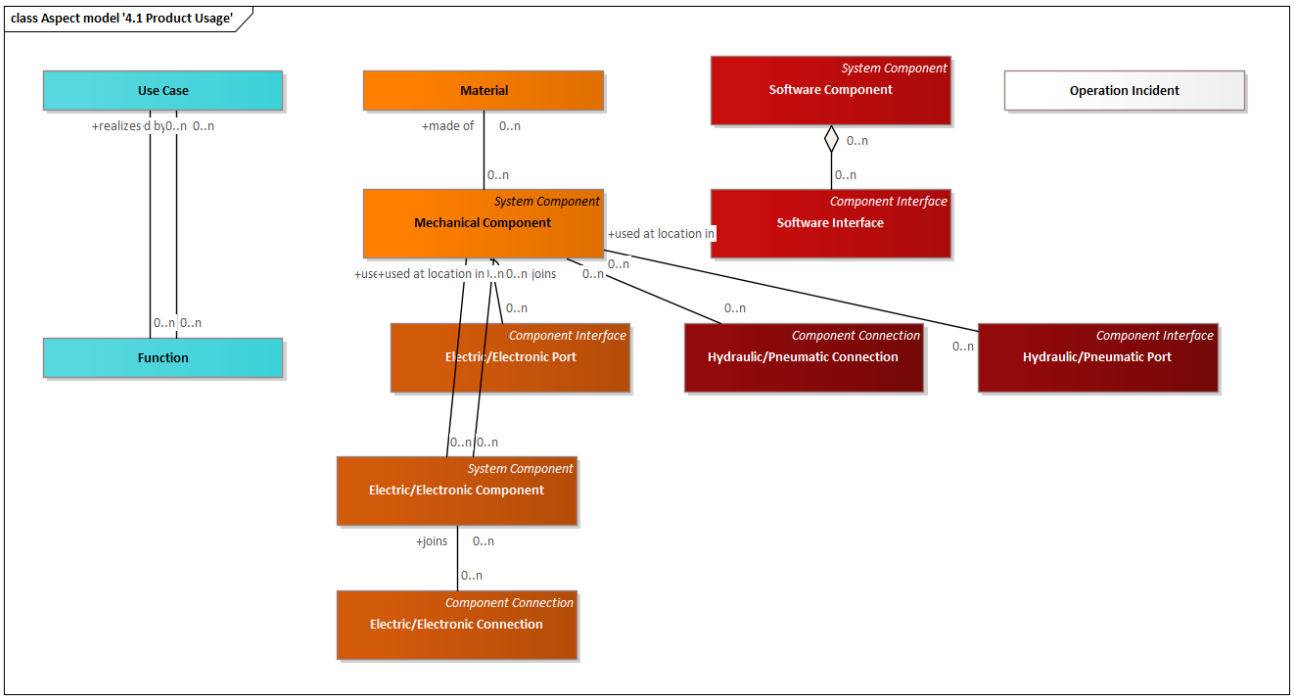


Figure 271 Aspect model '4.1 Product Usage'

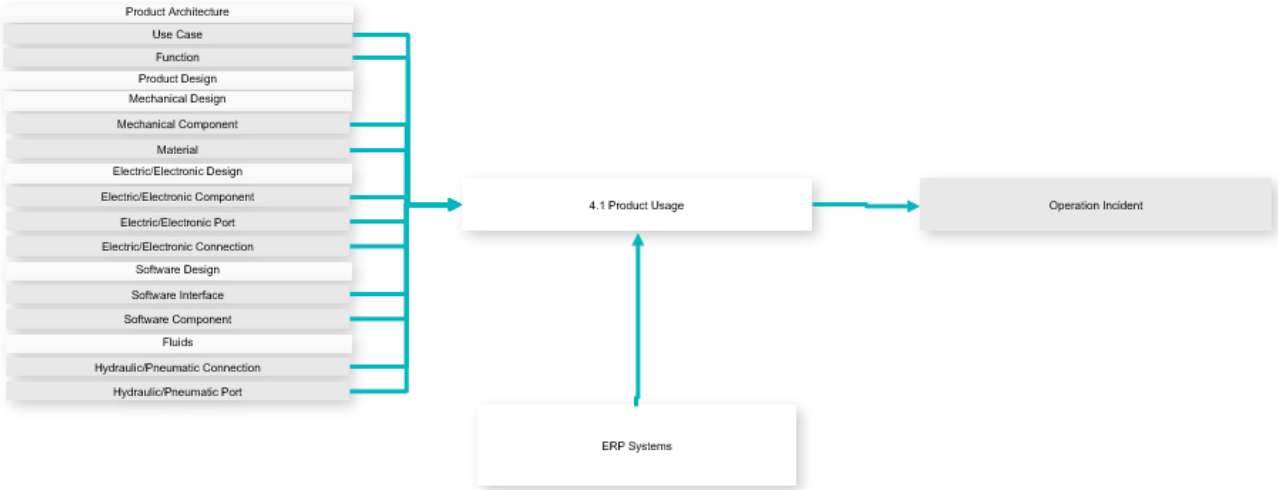


Figure 272 Use-case Diagram

Inputs	Use Case Function Mechanical Component Electric/Electronic Component Software Component Electric/Electronic Port Software Interface Hydraulic/Pneumatic Port Electric/Electronic Connection Hydraulic/Pneumatic Connection Material
Outputs	Operation Incident

8.5.5.2.1.6.2. Maintenance & Service

Includes all activities performed in order to maintain a product in operation.

Figure 273 Required Input

Figure 274 Used Tools

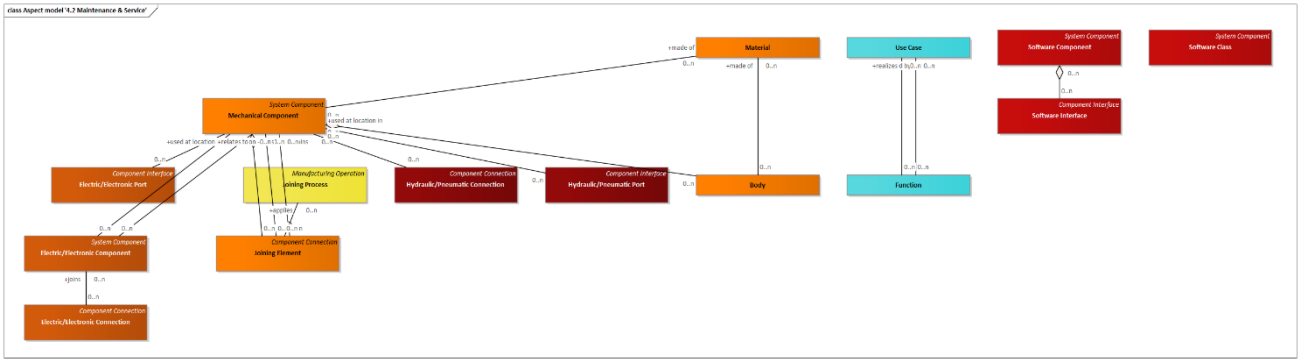


Figure 275 Aspect model '4.2 Maintenance & Service'

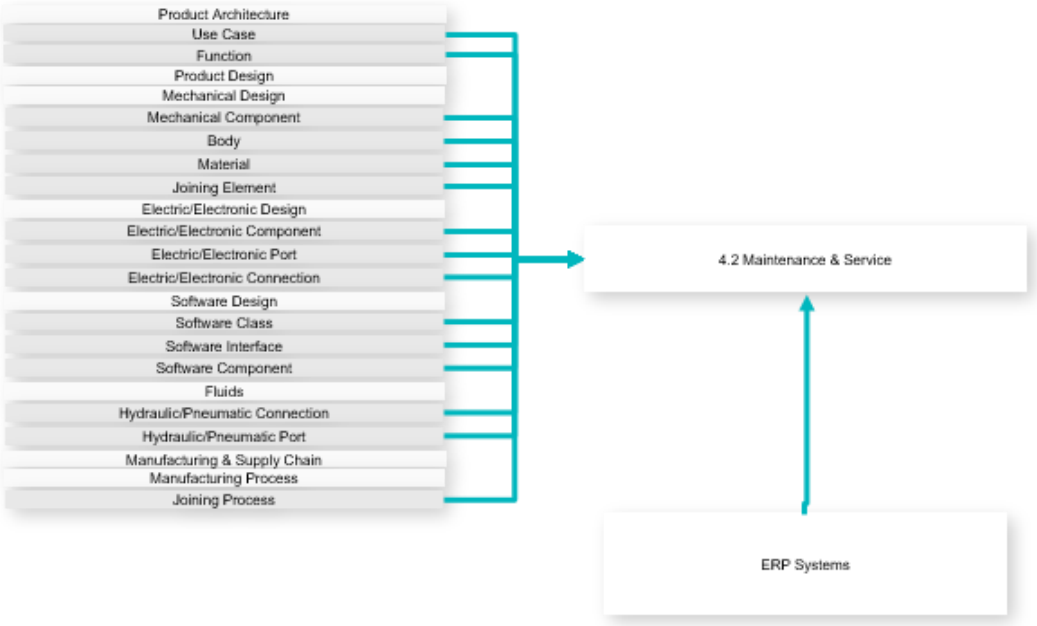


Figure 276 Use-case Diagram

Inputs	Use Case Function Mechanical Component Electric/Electronic Component Software Class Software Component Electric/Electronic Port Software Interface Hydraulic/Pneumatic Port Joining Element Electric/Electronic Connection Hydraulic/Pneumatic Connection Material Body Joining Process

8.5.5.2.1.6.3. Disassembly & Recycling

Includes all end-of-life activities required to putting a product or its component to recycling.

Figure 277 Required Input

Figure 278 Used Tools

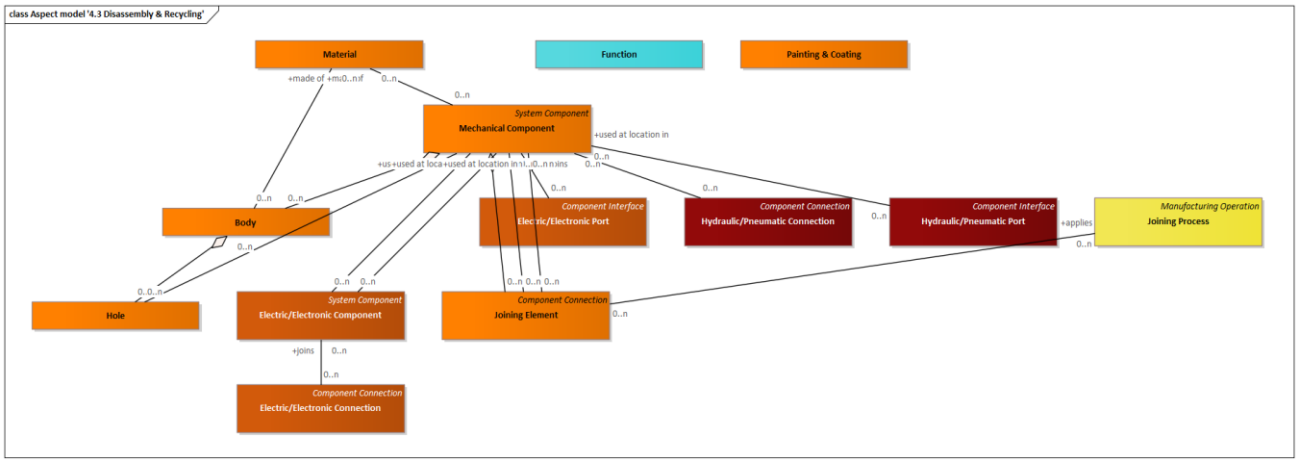


Figure 279 Aspect model '4.3 Disassembly & Recycling'

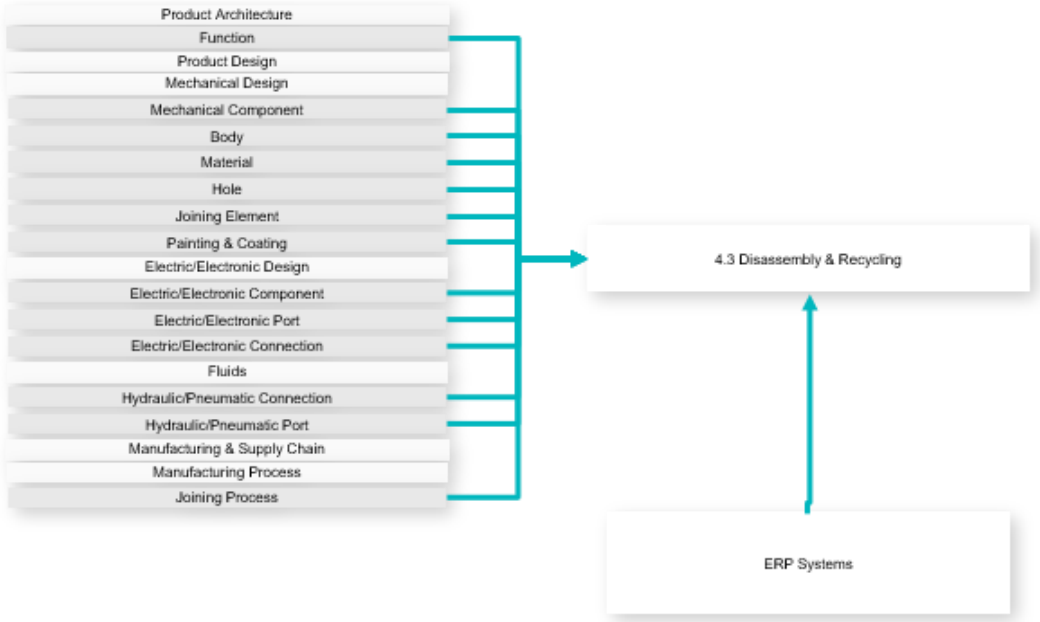


Figure 280 Use-case Diagram

Inputs	Function Mechanical Component Electric/Electronic Component Electric/Electronic Port Hydraulic/Pneumatic Port Joining Element Electric/Electronic Connection Hydraulic/Pneumatic Connection Material Body Hole Joining Process Painting & Coating

8.5.5.2.1.7. Procurement

Communication between procurement and suppliers in the bidding and iquiry phase. Supports the OMG Model-Based Aqcuisition initiative.

Figure 281 Required Input

Figure 282 Used Tools

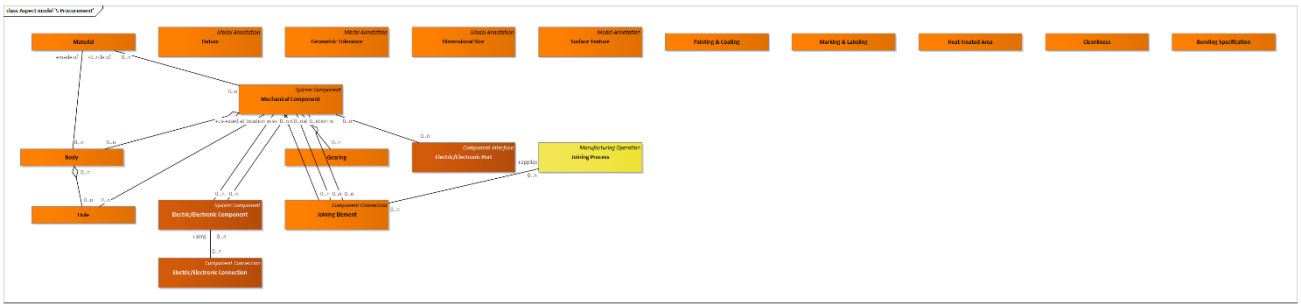


Figure 283 Aspect model '5 Procurement'

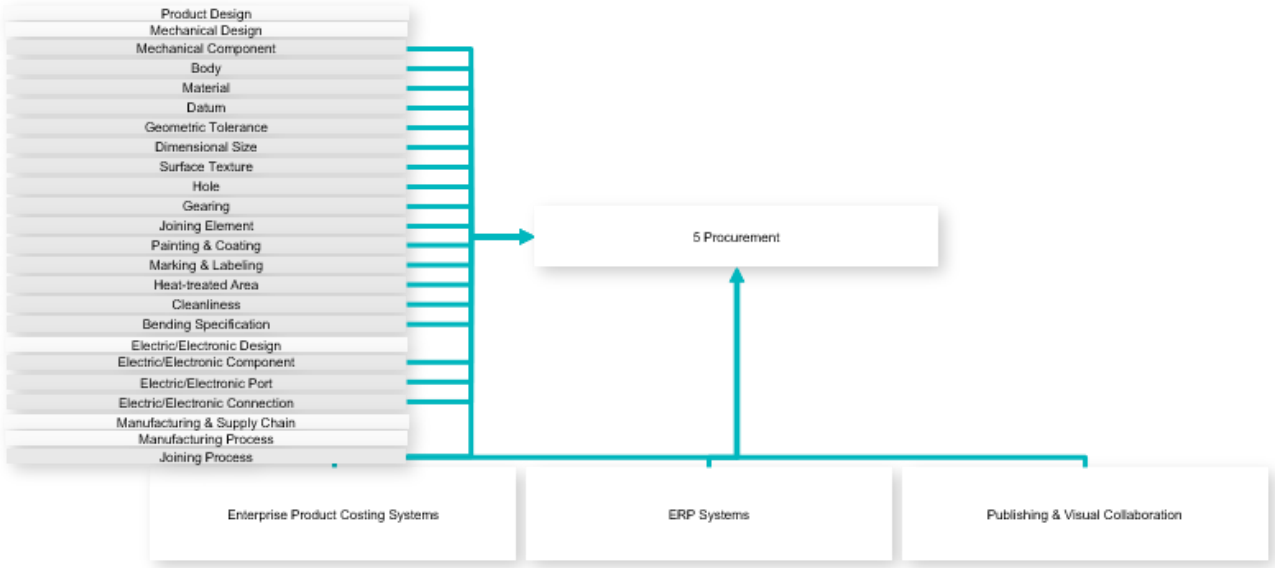


Figure 284 Use-case Diagram

Inputs	Mechanical Component Electric/Electronic Component Electric/Electronic Port Joining Element Electric/Electronic Connection Material Body Hole Gearing Datum Geometric Tolerance Dimensional Size Surface Texture Joining Process Painting & Coating Marking & Labeling Heat-treated Area Cleanliness Bending Specification

8.5.5.2.1.8. Product Line Engineering

Figure 285 Required Input

8.5.5.2.1.8.1. Design product line

Definition of the “shared asset supersets” and feature catalog: this is a collaborative effort typically involving the CU architect, customer product owner and possibly supplier tech marketing team. The goal is eventually to define the product family i.e. to identify all the assets and their potential variations that can be used to provide a solution to any given problem within the target problem domain, rather than building for each problem an ad-hoc solution. The problem domain is specified by the customer/product owner, but can be typically expanded by the tech marketing team who captures requirements from other customers and push for alignment with strategic objectives. In defining the product line the CU architect is driven by objectives as costs, reuse, IPs while the product owner wants the product family to allow for a problem solution as close as possible to an “optimal” ad-hoc architecture.

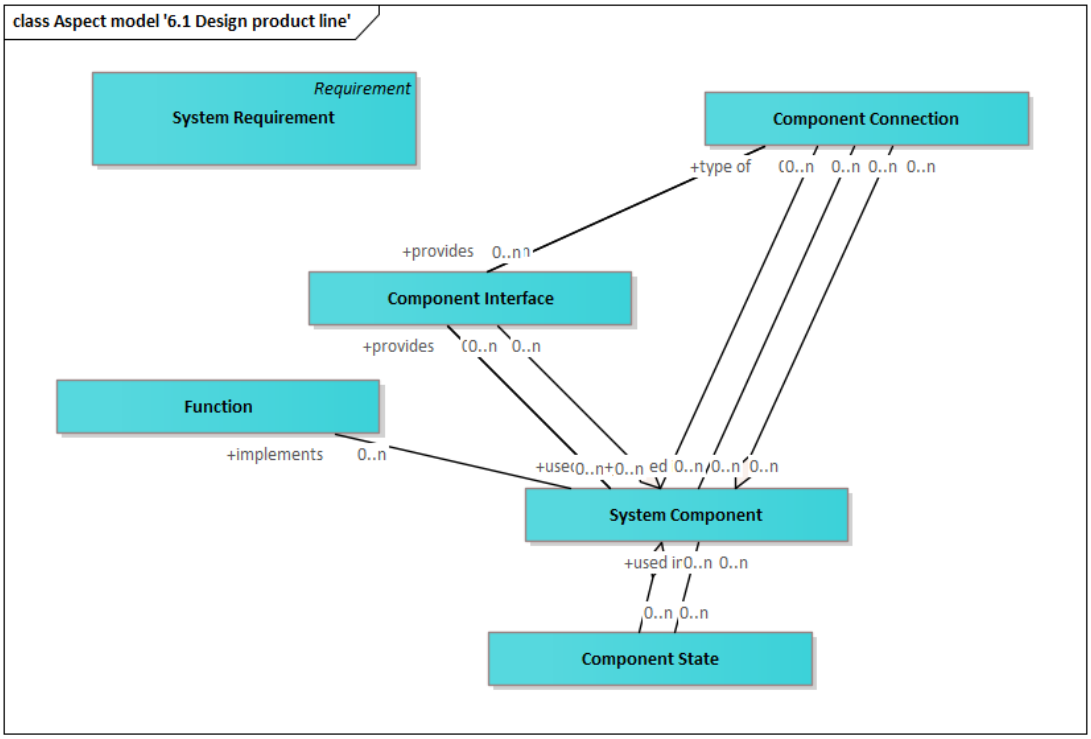


Figure 286 Aspect model '6.1 Design product line'

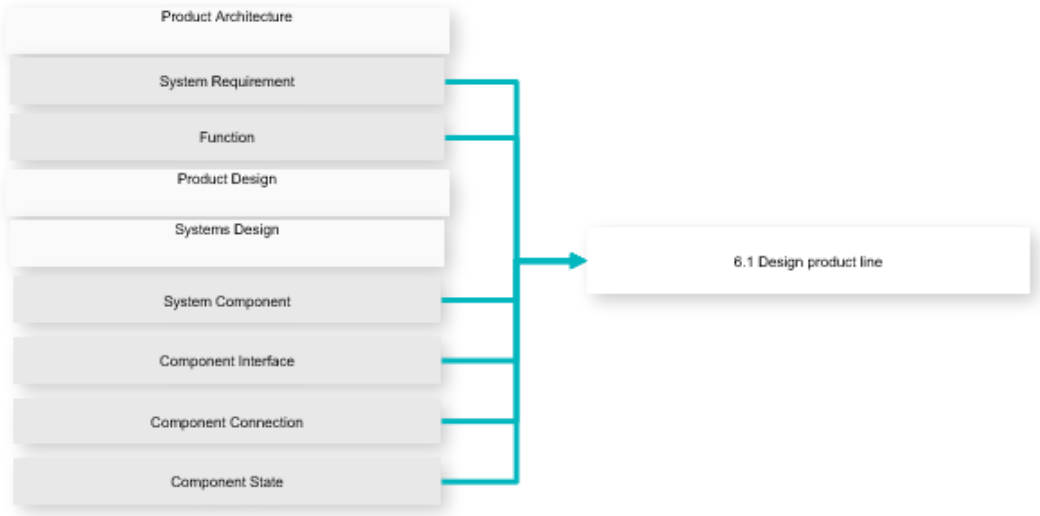


Figure 287 Use-case Diagram

Inputs	Function Component Interface System Requirement System Component Component Connection Component State

8.5.5.2.1.8.2. Design product asset instance

Design of the CU product asset instance, implying the definition of the bill of features: the ultimate goal of this collaboration use case is the specification of the features, selected from the feature catalogue, that define the solution to

the specific customer problem within the solution/family portfolio. The bill of feature is basically the set of actual values assigned to the variation point, which fully determine a single specific family instance. Collaboration still happens between product and CU architects, with the former specifying requirements and high level architecture for a specific problem (within their problem domain) and the latter providing a set of potential solutions, all belonging to the same CU family.

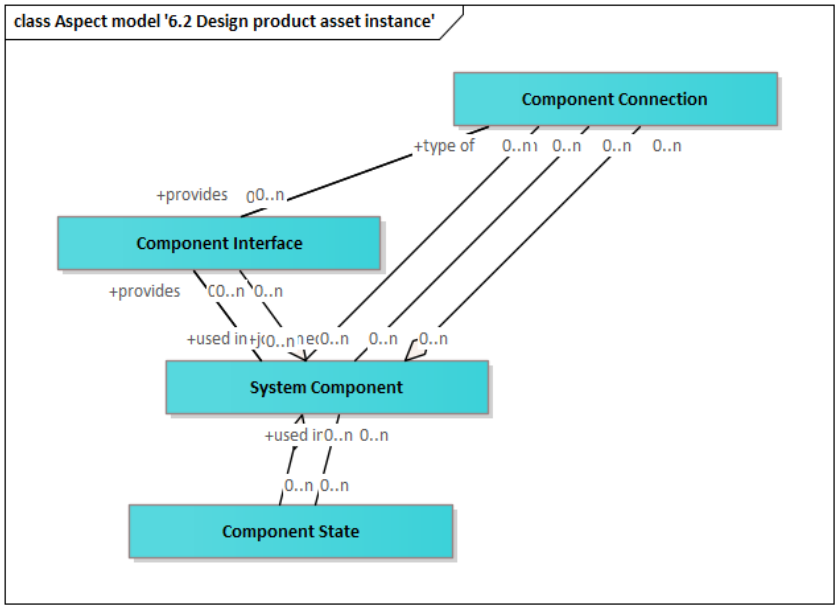


Figure 288 Aspect model '6.2 Design product asset instance'

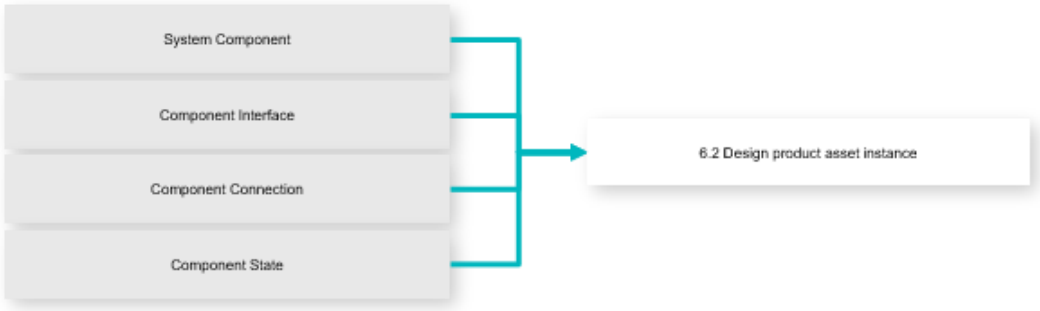


Figure 289 Use-case Diagram

Inputs	Component Interface System Component Component Connection Component State

8.5.5.2.1.8.3. Definition product asset instance configurability space

Definition of the CU product asset instance configurability space: the objective is to define the set of configuration options that will be made available to the CU end user for each CU instance. Typical options available for CU include clock selection, clock speed, caching and cache policy, protocol for external bus interfaces, general purpose I/O configuration (direction, active etc.) as well as dedicated options for peripherals and co-processors. A key aspect is also the definition of the configuration mechanisms that will be made available to the user including configuration protection features to prevent undesired/unauthorized changes. This is a collaborative efforts involving the same actors as Use Case 1, though driven by different criteria. The CU architect must balance the flexibility provided by configurability options with the added complexity it typically brings in terms of hardware design as well as verification effort. The product owner must balance programming complexity, potential higher cost and lower efficiency, with the capability of having their single CU instance capable of covering a broader problem space.

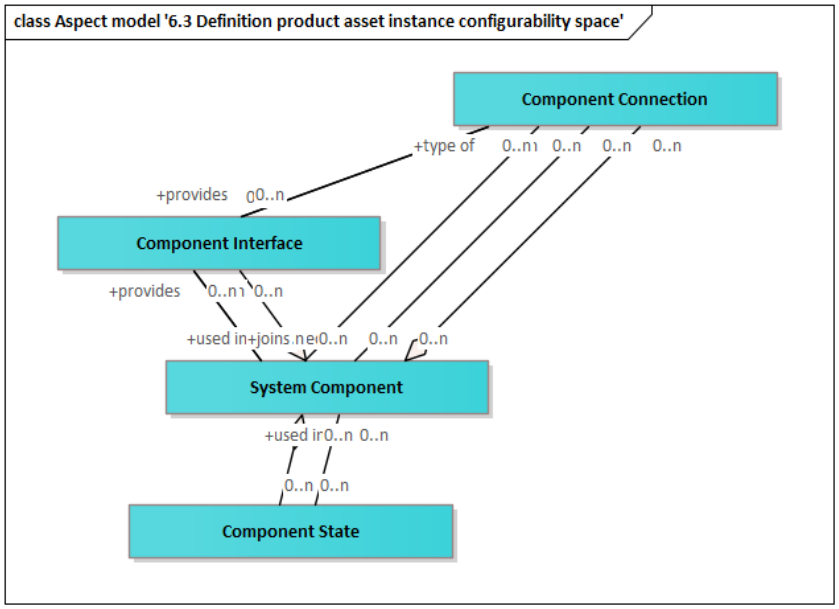


Figure 290 Aspect model '6.3 Definition product asset instance configurability space'

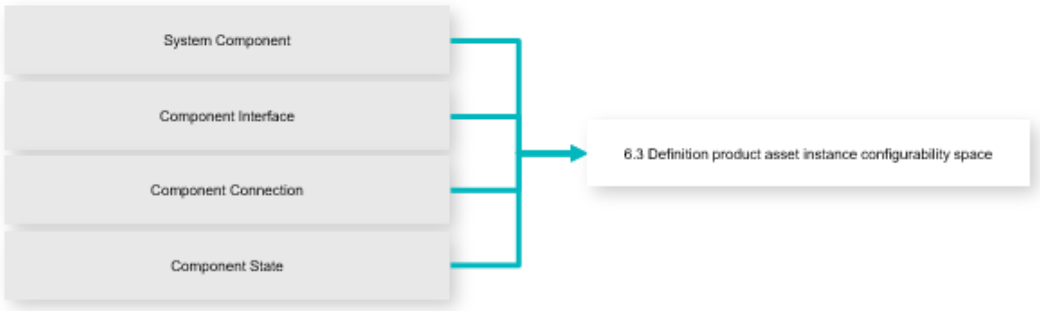


Figure 291 Use-case Diagram

Inputs	Component Interface System Component Component Connection Component State

8.5.5.2.1.9. Long-Term Archiving

For long-term archiving, all product information must be configured and stored in a retrievable way. Product information includes domain-specifc models, relations between models and model entities, and meta-data about the documents and models.

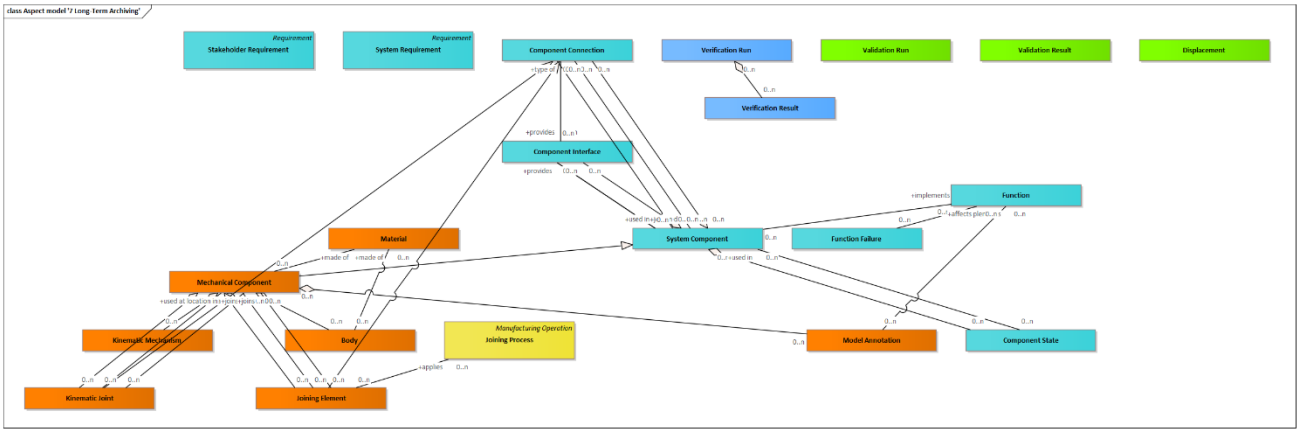


Figure 292 Aspect model '7 Long-Term Archiving'

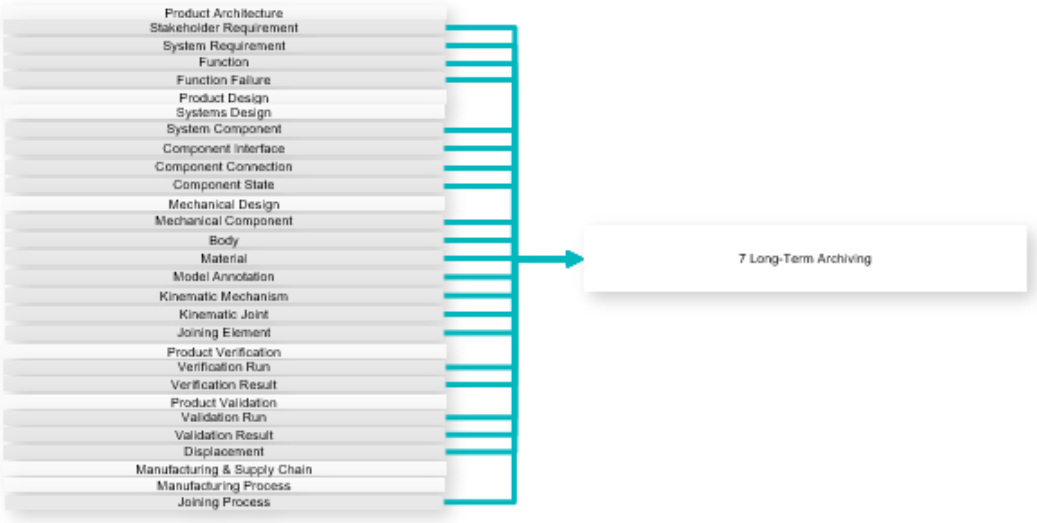


Figure 293 Use-case Diagram

Inputs	Function Component Interface Function Failure Stakeholder Requirement System Requirement System Component Model Annotation Mechanical Component Component Connection Component State Kinematic Joint Joining Element Material Body Kinematic Mechanism Joining Process Verification Run Verification Result Validation Run Validation Result

	Displacement

8.5.6. Related Standards

Lists all standards refered to from the classes of the ontology.

8.5.6.1. VDI 2519 Blatt 1 Vorgehensweise bei der Erstellung von Lasten-/Pflichtenheften

Related Classes	Use Case Requirement Function

8.5.6.2. CSA ISO/IEC/IEEE 15288 System life cycle processes

Related Classes	Requirement Function Stakeholder Requirement System Requirement System Component

8.5.6.3. ISO/IEC 20246 Bewertungen von Arbeitsergebnissen

Related Classes	Function Component Interface Function Failure System Requirement System Component Software Class Software Component Component Connection Component State Software Interface

8.5.6.4. ISO/IEC/IEEE 42010 Software, systems and enterprise - Architecture description

Related Classes	System Component

8.5.6.5. ISO/IEC/IEEE 29148 Requirements Engineering

6.4.3.2 Perform validation

This activity consists of the following task: ISO29148

NOTE Tasks 1), 3), 4), and 5) under this activity are not included, as there is no specific guidance related to requirements engineering.

2) Conduct validation to demonstrate conformance of services to stakeholder requirements.

NOTE Validation is undertaken in a manner, consistent with organizational constraints, such that uncertainty in the replication of validation actions, conditions and outcomes is minimized. Objectively record and approve validation actions and results. Validation may also be conducted to confirm that the system not only satisfies all operational, functional and usability requirements, but also satisfies the often less formally expressed, but sometimes overriding, attitudes, experience and subjective tests that comprise customer satisfaction.

[ISO/IEC 15288:2008 (IEEE Std 15288-2008), 6.4.8.3 b) 2)]

Figure 294 Validation

6.4.2.2 Perform verification

This activity consists of the following tasks: ISO29148

NOTE Tasks 1), 3), and 4) under this activity are not included, as there is no specific guidance related to requirements engineering.

> 2) Conduct verification to demonstrate compliance to the specified design requirements.
>
> NOTE Non-compliance identifies the existence of random faults and/or design errors, and corrective actions are initiated as appropriate. Verification is undertaken in a manner, consistent with organizational constraints, such that uncertainty in the replication of verification actions, conditions and outcomes is minimized. Approved records of verification actions and outcomes are made.
>
> *[ISO/IEC 15288:2008 (IEEE Std 15288-2008), 6.4.6.3 b) 2)]*

Figure 295 Verification

Related Classes	Requirement

8.5.6.6. ISO 26262 Funktionale Sicherheit

Related Classes	Function Failure

8.5.6.7. ISO 21448 Safety of the intended functionality

Related Classes	Function Failure

8.5.6.8. ARP4754B Guidelines for Development of Civil Aircraft and Systems

Related Classes	Function Failure

8.5.6.9. ARP4761A Guidelines for Conducting the Safety Assessment Process on Civil Aircraft, Systems, and Equipment

Related Classes	Function Failure

8.5.6.10. DIN EN 61508-1 Part 1: General requirements

Related Classes	Function Failure

8.5.6.11. VDA 231-200 Werkstoffdatensatz - Spezifikation von Werkstoffen und Oberflächen in IT-Systemen

Für die Entwicklung und Produktion von Fahrzeugen ist die Automobilindustrie zunehmend auf elektronische Produktdaten angewiesen. Die Produktdaten beschreiben geometrische Daten, Stücklisten, Ablaufpläne, Freigabestatus, Produktstruktur und Fertigungsdaten von der Konzeption bis zur Fertigung und bilden die Basis für Bereiche wie CAD, CAM, CAE und PDM.
Werkstoffdaten zu Sachnummern sind originäre Produktdaten und somit wesentlicher Bestandteil des Produktdatenmanagement. Eindeutige Werkstoffangaben sind Voraussetzung für Prozesse wie Konzeption, Berechnung, Versuch, Zeichnungserstellung, Fertigungsplanung, Beschaffung, Logistik, Typzulassung und Recycling. Werkstoffdaten werden deshalb zwischen verschiedenen IT-Systemen ausgetauscht, sowohl firmenintern als auch mit Zulieferern und Halbzeugherstellern.
Zweck dieser Spezifikation ist die Definition eines einheitlichen Datensatzes, der es ermöglicht Werkstoffe IT-technisch so eindeutig zu spezifizieren, wie es für die technische Produktdokumentation auf der Zeichnung (DIN EN ISO 10209, ISO 128-1) erforderlich ist. Die Werkstoffangabe beschreibt den Werkstoff im Endzustand des Produktes für eine bestimmte Funktion (ISO 128). Diese Angaben können durch entsprechende Kennzeichnung eine Dokumentationspflicht der Bauteileigenschaften auslösen. Der Werkstoff muss soweit detailliert spezifiziert sein, dass mit diesen Angaben reproduzierbare Bauteileigenschaften gewährleistet sind und die Anforderungen bei der Herstellung überprüfbar und erfüllbar sind.

Diese Spezifikation soll eine Grundlage für zukünftige IT-Systeme bilden. Sie kann für den Datenaustausch zwischen Kunden und Lieferanten vereinbart werden.
Es ist nicht vorgesehen, den hier beschriebenen Datensatz kurzfristig generell verbindlich vorzuschreiben, er kann aber zwischen Auftraggeber und Lieferant individuell vereinbart werden.
Werkstoffspezifikationen können in dieser Form in einem zentralen / führenden IT-System abgelegt werden.
Weiterführende IT-Systeme können diese Werkstoffangabe zur eindeutigen Identifikation nutzen und bei Bedarf darauf referenzieren.
Neben der Beschreibung der einzelnen Feldinhalte in Kapitel 3 enthält diese Spezifikation im Beiblatt 1 die vorgegebenen Pfadschlüssel der Werkstoffreferenz in den Feldern 17 bis 21. Weiterhin wird in Kapitel 4 ein Datenaustauschformat zur Übertragung der Daten zwischen verschiedenen IT-Systemen zur Verfügung gestellt. Ein entsprechendes Listing enthält Beiblatt 2. Zusätzliche Anwendungsbeispiele sind im Anhang dargestellt.

Related Classes Material

8.5.6.12. VDA 231-106 Werkstoff-Klassifizierung im Kraftfahrzeugbau: Aufbau und Nomenklatur

Die Klassifizierung nach dieser Empfehlung gilt für Werkstoffe und Werkstoffverbunde in Kraftfahrzeugen, deren Anhängern sowie für Teile, die für diese Fahrzeuge verwendet werden.
Sie stützt sich auf die herkömmlichen Bezeichnungen der Werkstoffe; nach Bedarf werden auch neue Bezeichnungen verwendet.
Die Klassifizierung dient der eindeutigen Zuordnung der Werkstoffe und Werkstoffverbundteile in eine überschaubare Struktur und folgenden Zwecken:
- die Vergleichbarkeit unterschiedlicher Fahrzeuge in Hinblick auf die verwendeten Werkstoffe
- die Ermittlung von Werkstoffanteilen im Hinblick auf die gesetzlich vorgeschriebenen Quoten für die Kraftfahrzeugverwertung
- die Einstufung der Werkstoffangaben aus dem „Materialdatenblatt“ für Bauteile der Lieferanten

Related Classes Material

8.5.6.13. ISO 7499 Unique integral feature identification (UIFI)

Related Classes Face

8.5.6.14. DIN EN ISO 1101 Tolerierung von Form, Richtung, Ort und Lauf (URL-Dublette!)

Related Classes Geometric Tolerance

8.5.6.15. DIN EN ISO 14405 Teil 2: Andere als lineare oder Winkelgrößenmaße (ISO 14405-2:2018)

Related Classes Dimensional Size

8.5.6.16. DIN EN ISO 14405 Teil 3: Winkelgrößenmaße (ISO 14405-3:2016)

see: DIN EN ISO 14405 Teil 3: Winkelgrößenmaße (ISO 14405-3:2016)

Related Classes Dimensional Size

8.5.6.17. DIN EN ISO 14405 Teil 1: Lineare Größenmaße (ISO 14405-1:2016)

see: DIN EN ISO 14405 Teil 1: Lineare Größenmaße (ISO 14405-1:2016)

Related Classes Dimensional Size

8.5.6.18. DIN EN ISO 21920 Teil 1: Angabe der Oberflächenbeschaffenheit (ISO 21920-1:2021)

Related Classes Surface Texture

8.5.6.19. DIN EN ISO 21920 Teil 3: Spezifikationsoperatoren (ISO 21920-3:2021)

see: DIN EN ISO 21920 Teil 3: Spezifikationsoperatoren (ISO 21920-3:2021)

Related Classes Surface Texture

8.5.6.20. DIN EN ISO 21920 Teil 2: Begriffe und Kenngrößen für die Oberflächenbeschaffenheit (ISO 21920-2:2021, korrigierte Fassung 2022-06)

see: DIN EN ISO 21920 Teil 2: Begriffe und Kenngrößen für die Oberflächenbeschaffenheit (ISO 21920-2:2021, korrigierte Fassung 2022-06)

Related Classes	Surface Texture

8.5.6.21. DIN ISO 15786 Vereinfachte Darstellung und Bemaßung von Löchern (ISO 15786:2008)

Related Classes	Hole Gearing

8.5.6.22. DIN 8580 Begriffe, Einteilung

Related Classes	Heat Treatment

8.5.6.23. DIN EN ISO 4885 Eisenwerkstoffe - Wärmebehandlung - Begriffe

Related Classes	Heat-treated Area

8.5.6.24. ISO 16232 Cleanliness of components and systems

Related Classes	Cleanliness

8.5.6.25. ISO/IEC 19513 Information technology - Object Management Group Unified Profile for DoDAF and MODAF (UPDM)

Related Classes	Software Class

8.5.6.26. DO-297 Integrated Modular Avionics (IMA) Development Guidance and Certification Considerations

Related Classes	Software Component Software Interface

8.5.6.27. ISO/IEC/IEEE 29119-1 Software and systems engineering - Software testing - Part 1: General concepts

Related Classes	Validation Criterium Software Component Software Interface

8.5.6.28. DIN 23601 Verification - Requirements for measurements to determine the characteristics for size, form, orientation, location and run-out

Related Classes	Verification Characteristic Verification Resource Verification Plan

8.5.6.29. ISO 7533 Identification of specifications in the technical product documentation (TPD)

Related Classes	Verification Characteristic

8.5.6.30. VDA 231-300 Digitaler Datenaustausch in der werkstofflichen Bemusterung

8.5.6.31. unter Berücksichtigung der 3D-Daten

Related Classes	Verification Characteristic

8.5.6.32. ISO 22514 Capability and performance

Related Classes	Verification Result

8.5.6.33. ISO 20170 Decomposition of geometrical characteristics for manufacturing control

Related Classes	Verification Result

8.5.6.34. ISO 26580 Software and systems engineering - Methods and tools for the feature-based approach to software and systems product line engineering

Related Classes	Configuration Variant

8.6. CASCaRA Application Programming Interface (API)

This section is TBD

8.7. CASCaRA Transformations

This section is TBD