 THE RESEARCH LEADING TO THESE RESULTS HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S SEVENTH FRAMEWORK PROGRAM (FP7/2007-2013) FOR CRYSTAL – CRITICAL SYSTEM ENGINEERING ACCELERATION JOINT UNDERTAKING UNDER GRANT AGREEMENT N° 332830 AND FROM SPECIFIC NATIONAL PROGRAMS AND / OR FUNDING AUTHORITIES.

CRitical S**YST**em Engineering **A**cce**L**eration

State of the art for Aerospace ontology

D209.010

DOCUMENT INFORMATION

Project	CRYSTAL
Grant Agreement No.	ARTEMIS-2012-1-332830
Deliverable Title	State of the art for aerospace ontology
Deliverable No.	D209.010
Dissemination Level	PU
Nature	R
Document Version	V1.0
Date	2014-01-30
Contact	Ivo Viglietti
Organization	ALA
Phone	+39 011 756 3191
E-Mail	ivo.viglietti@alenia.it

AUTHORS TABLE

Name	Company	E-Mail
Ivo Viglietti	ALA	Ivo.viglietti@alenia.it
Anne Monceaux	EADS-IW	Anne.Monceaux@eads.net
Federico Tomassetti	PoliTO	federico.tomassetti@polito.it

REVIEWER

Version	Date	Reviewer
V0.8	2014-01-10	Anne Monceaux
V0.9	2014-01-17	Michael Van Bekkum, Nicolas Figay

CHANGE HISTORY

Version	Date	Reason for Change	Pages Affected
0.01	2013-07-22	First document layout	All
0.02	2013-10-16	EADS contribution	All
0.03	2013-11-25	Section on partners SoA added Added info about view point concepts Added info about CESAR outcomes and STEP	3 4
0.04	2013-12-02	Section 2 updated and completed Added section 7.1	

Version	Date	Reason for Change	Pages Affected
0.05	2013-12-05	Section 3.1 updated with ALA contribution Annex IV added	
0.06	2013-12-05	Text updated in the section explaining relations to other projects results Section 6.2 name changed Added contributions to Semantic WEB technologies	1.3 6 7.1
0.07	2014-01-08	Added contributions from EADS-IW Added description for Protégé and Virtuoso Added content to PLCS OWL mapping	All 7 Annex III
0.08	2014-01-10	Version for first level review	All sections
0.09	2014-01-17	Version for second level review	All sections
1.0	2014-01-29	Final version	All sections

CONTENT

1 INTRODUCTION 8

1.1 ROLE OF DELIVERABLE 8
1.2 EXECUTIVE SUMMARY 8
1.3 RELATIONSHIP TO OTHER CRYSTAL DOCUMENTS 9
1.4 RELATIONS WITH OTHER PROJECTS RESULTS 10
1.5 STRUCTURE OF THIS DOCUMENT 10

2 ONTOLOGY RELATED CONCEPTS DEFINITION 12

2.1 ONTOLOGY 12
2.2 ROLE OF DOMAIN ONTOLOGIES FOR CRYSTAL IOS 14
2.3 SCOPE OF THE DOMAIN ONTOLOGY 15
2.3.1 Ontology coverage in synthesis 17
2.3.2 SE concepts in aerospace domain 17
2.3.3 Reference data & information shared in SE process 18
2.3.4 Specific application resources 19

3 PARTNER SOA REGARDING SE CONCEPTS AND REFERENCE DATA MODELING 20

3.1 INDUSTRIAL PARTNERS SOA 20
3.2 ONTOLOGY FOR REQUIREMENT CONTROLLED AUTHORING 26

4 STANDARD FOR ENGINEERING DATA REPRESENTATION 30

4.1 OVERVIEW 30
4.2 REQUIREMENTS DATA 32
4.3 SYSTEM BREAKDOWN 33
4.3.1 Defining viewpoints 35
4.4 PRODUCT STRUCTURE AND CONFIGURATION 39

5 STANDARD FOR ENGINEERING DEVELOPMENT PROCESS 42

5.1 PRODUCT LIFECYCLE SUPPORT 42
5.1.1 The PLCS Initiative 42
5.1.2 DEX 45
5.2 MANAGING COLLABORATION PROCESSES 46

6 ONTOLOGY NEEDS FOR REQUIREMENTS ENGINEERING AND V&V 48

6.1 OVERVIEW 48
6.2 REQUIREMENT MODELING VS TRACEABILITY BETWEEN REQUIREMENTS AND MODELS 49
6.3 SEMI-FORMAL REQUIREMENTS SPECIFICATION AND V&V 52

7 SUPPORT TOOLS AND TECHNOLOGIES 57

7.1 SEMANTIC WEB TECHNOLOGIES 57
7.1.1 W3C Standards 57
7.1.2 Formalisation of knowledge 60
7.2 COLLABORATIVE PLATFORMS 71
7.2.1 W3C technologies 71
7.2.2 Current / envisioned usage of ontology in RTP platform 74

8 TERMS, ABBREVIATIONS AND DEFINITIONS 76
9 SHORT LITERATURE REVIEW 77
10 REFERENCES 79
11 ANNEXES 82

CONTENT OF FIGURES

Figure 2-1 Different knowledge areas in CRYSTAL 12
Figure 2-2: Preliminary vision of the Domain Ontology context 16
Figure 2-3: Layered scope for a domain ontology 17
Figure 3-1: Conceptual model of the Functional view 21
Figure 3-2: Approach for the Traceability View 22
Figure 3-5: Virtual aircraft model overview 23
Figure 3-6: EDM main concepts 24
Figure 3-7 – BDA model overview 25
Figure 3-6: REUSE KM Thesaurus management 27
Figure 3-7: SKOS representation of RQS ontology 28
Figure 3-8: Representation Schemas 29
Figure 4-1: AP233 vs. AP239 overlaps 31
Figure 4-2: AP233 Requirements concept structure 32
Figure 4-3: Breakdowns concepts 33
Figure 4-4: Breakdowns structure in AP233 34
Figure 4-5: View model 36
Figure 5-1: PLCS Standard definition according to “Eurostep” 43
Figure 5-2: AP239 Capabilities 44
Figure 5-3: PLCS concept model 45
Figure 5-4: OASIS DEX architecture 46
Figure 6-1: UML state machine diagram 49
Figure 6-2: IHM screenshot of COGNITION cockpit 54
Figure 6-3: TXM, an XML-TEI encoded corpora compatible analysis platform for text mining 55
Figure 6-4: Retrieving parametrized information within a requirement 56
Figure 7-1: DBpedia concept 59
Figure 7-2: Web Technology Stack 61
Figure 7-3: GUIs examples from Protégé editor 73
Figure 7-4: using OpenLink Virtuoso 74

Figure 7-5: Virtuoso general architecture 75

Content of Tables

Table 1 - Main concerns handled by viewpoints 39
Table 7-1: Terms, Abbreviations and Definitions 76

Content of Appendixes

ANNEX I: DEX SPECIFICATIONS **82**
ANNEX II: OSLC CONCERNS **83**
ANNEX III: PLCS – AN OWL MAP **84**
ANNEX IV: ALA FUNCTIONAL VIEW DATA DICTIONARY **193**
ANNEX V: BDA PACKAGE OVERVIEW **197**

1 Introduction

1.1 Role of deliverable

The objective of this document is to present a comprehensive picture on the practices, methods and tools that we consider as starting status to treat the product data description in the aerospace domain in CRYSTAL.

1.2 Executive summary

At this stage of the Work package progress, we have identified and agreed on three main “pillars” for the CRYSTAL Aerospace domain ontology construction, and one “direction” for improving the coverage and formality degree of ontological descriptions.

The three “pillars” address the driving needs to identify and describe the abstract classes of objects to be exchanged in the project MBSE use cases. They are:

- the STEP standards (AP233, AP239) that we consider, provide the basis vocabulary and extension mechanisms to extend the IOS/OSLC core vocabulary
- the Semantic web formalisms (RDFS, OWL) that have been adopted within the IOS OSLC specification
- integrated tool support for edition, query, etc. (SPARQL, PROTÉGÉ, VIRTUOSO). This architecture and integration with IOS must be specified in specification task T209-02.

The direction for improvement concerns more specifically the description of concepts common to Requirement engineering and Functional architecture domains, such as the “key value types” or parameter types (e.g. condition parameters, function parameters, etc…), that define the significant values that need to be exposed, e.g. for display on dashboards, for checks, or for monitoring and control. This appears to be needed to enable proper implementation of many use cases.

After introducing the concept of ontology and summarizing what is considered the role, scope and possible coverage of a Domain ontology in CRYSTAL, the document provides a survey about existing referential in aerospace domain which can be used as a starting point for building the SP2 domain ontology. Existing standards, mainly STEP and de facto standards and practices (e.g.

procedures between manufacturer and suppliers) to share common information in aerospace are pointed out and discussed. The outcomes of previous research projects (e.g. CESAR, CRESCENDO) dealing with the objective of representing and sharing common engineering data are analyzed.

A chapter argues that the domain ontology should entail descriptions of the key parameters to deal with requirements in MBSE and other uses cases in the project.

Finally the document includes a description of RDF based languages and semantic web technologies usable in the project.

1.3 Relationship to other CRYSTAL Documents

This document will serve as a basis to Tasks 209.2 and 209.3.

The Aerospace domain ontology boundaries will depend on the expressed industrial needs. This link is established by referencing the Aerospace Common Use Case. This implies considering the inputs from work package WP208 about Aerospace use case description which will illustrate on a public use case the several engineering methods defined by the SP2 use cases to express their needs (

|Kyo, 1990]	Kyo C. Kang and al. Feature-Oriented Domain Analysis (FODA), 1990.
IWSSD '89	Proceedings of the 5th international workshop on Software specification and design 152-159. 1989
[Moros, 2008]	Begoña Moros, Cristina Vicente-Chicote, Ambrosio Toval, *Metamodeling Variability to Enable Requirements Reuse, in* Proceedings of EMMSAD 2008
Siegmund,	Norbert Siegmund, Martin Kuhlemann, Marko Rosenmuller, Christian Kaestner, and Gunter Saake, Integrated Product Line Model for Semi-Automated Product Derivation Using Non-Functional Properties, in
CRYSTAL D208.010	CRYSTAL aerospace use case description Report – V1 D208.010, 2014.

).

At the same time, a strong link with the RTP platform is envisioned. The domain ontology should be a resource usable by CRYSTAL RTP services to fulfil different types of needs corresponding to those expressed by the CRYSTAL aerospace partners use case scenarios.

In this respect, an exchange of information with WP601 is needed. The work performed on other CRYSTAL domains ontologies in work-packages 3.8, 4.7 and 5.4, shall also be considered in order to identify potential commonalities and standardization issues.

1.4 Relations with other projects results

The concepts and ideas that are analyzed within previous and on-going projects will be considered here. As already stated before, ARTEMIS projects CESAR and MBAT deal with Interoperability and data representation issues: their findings are considered with reference to many topics:

- Formal Languages
- Data models
- Process specification
- Requirements specification
- Usage scenarios

The VIVACE and CRESCENDO projects are also considered when dealing with Aircraft's modeling and simulation data objects and enterprise level collaboration.

As part of companies own background related to ontology, some experiences and baseline information are reported. This includes past collaborations with academic research institutes and nationally funded research initiatives, such as the "iDesign foundation" Italian project.

1.5 Structure of this document

This document is organized as follows:

- **Section 1** provides an overview of the goals of the document, of the objectives of task T209.1 and of the relationship with other documents and previous projects.
- **Section 2** recalls the definitions agreed in the context of CRYSTAL for the most significant terms and concepts adopted to address the domain ontology topic.
- **Section 3** reports partners existing knowledge about ontology.
- **Section 4 and 5** report the results of our survey about STEP standards and concepts available for representing process and product lifecycle related information.
- **Section 6** analyses possible existing approaches for using ontology in requirements engineering.

- **Section 7** analyses the available tools and technologies for supporting the integration of domain ontologies within the industrial cases.
- **Section 8** lists the abbreviations adopted within this document.
- **Section 9** reports the referenced documents.

2 Ontology related concepts definition

This section introduces the concept of ontology and summarizes what is considered the role, scope and possible coverage of a Domain ontology in CRYSTAL.

2.1 Ontology

The term ontology is used differently by different communities and may designate different kinds of artifacts. In CRYSTAL, it is actually the case that different kinds of ontologies are present or foreseen for describing Modeling languages concepts, System Engineering concepts and Disciplines vocabulary, respectively. Below we define them.

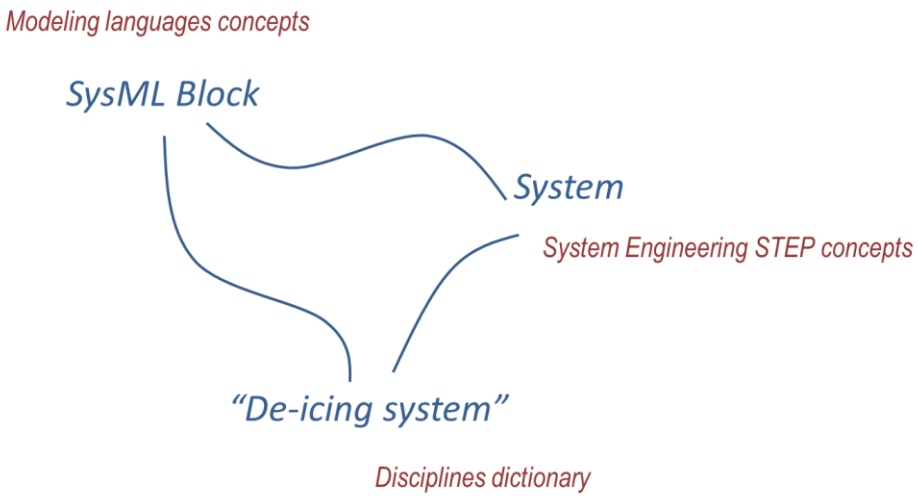


Figure 2-1 Different knowledge areas in CRYSTAL

In philosophy the Ontology is the study of "the nature of being". Its object is the study of what exists, and of the general properties of what exists.

Outside philosophy, ontology is used in a different, narrower meaning: an ontology is a **description** of what exists, usually in a particular domain or for a particular application. It focuses on naming things and grouping similar things into categories. It gives an explicit formal specification of the concepts in a domain and relations among them. The names of the concepts and their definition in natural language are important features for the ability to understand and use an ontology. Scope, completeness properties are depending on the intent and foreseen application: for Information Retrieval purpose one can emphasize the largest coverage ("what exists" in the domain?); but in the field of Artificial Intelligence, people are more interested in the

concepts required for useful reasoning in a domain, and not so much in the question whether a concept exists in the physical world or not.

Finally, when this description is formalized using a particular language, the term “ontology” also designates the obtained artifact. Such artifacts are designed either for human shared understanding of the concepts, or for machine processing. Depending on the language expressivity and properties, different algorithms may apply to it and authorize automatic processing and reasoning.

The above definitions require additional comments since the term is used by different communities and may designate different kinds of artifacts depending on the degree of formalization, the scope, or the intent in terms of application. For the sake of brevity we consider below four different types.

- **Upper-level ontologies** are theories that define abstract categories by means of spatial, temporal, causal, part-hood, etc. properties suitable to impose a structure on large lexical repositories [Guarino:98] or on lower level domain ontologies.
- **Domain ontologies** are agreed specifications of how to describe the concepts of particular domains of interest; they are merely being applied within the areas of data management and information sharing. In Crystal, an ontology of System Engineering describes the concepts of the SE domain such as *“System”, “Operation”, “Function”*, etc. An example of an OWL representation of the SE concepts defined by the STEP standards is presented in this document.
- **Linguistic ontologies** characterize unambiguously the meaning of some linguistic expressions for a given context determined by the ontology intended application [Bachimont,00]. They define various relationships between words or between terms depending again on the intended usage (e.g. taxonomic relations (broader), synonymy, normative (preference), associations, etc.). In CRYSTAL, the “KM ontology” built in 6.7 and 2.4 RBE activities and intended to be used to control the authoring quality of the requirements pertains to this category (see below § 3.2). It describes and structures relations between terms like “*Ice protection system”, “De/anti icing system”, “De-icing system*”, *etc.* that are used by people in various documents. The “KM ontology” is integrated inside the RQS tool of REUSE Company.
- **Application ontologies**. Technically, OSLC deals with the integration of heterogeneous legacy tools - using different languages - by means of Semantic Mediation technologies

that include application ontologies. In the area of Semantic mediation an ontology is usually an abstract model of the common semantics of the to-be-integrated tools metamodels. Assuming that common "concepts" are shared by multiple modelling tools, a semantic mediation is a mechanism to map the tool1 concepts with their tool2 equivalents. To illustrate this, we can take a simple example. A "*System Component*" SE concept can be represented by a "Block" in SysML, a "Class" in Modelica and so on. In this case, the "*System Component*" is a higher abstraction level concept that is realized differently in the different tools/languages; and a "Block" is a lower abstraction level SysML concept. Partial equivalences can be represented between tools concepts. In case the several tool concepts are modeled with different sets of attributes and properties; it is possible to define a semantic element that sets the minimal shared definition. This principle is a basis of the OSLC project. The implementation of a mediation mechanism may require several representation layers for respectively describing the language elements of each tool and the rules to map them. CRYSTAL follows up on past EU projects such as SPEEDS, CESAR, SPRINT or DANSE. Example implementation of OSLC ontologies is described in ([DANSE, 2012]).

2.2 Role of domain ontologies for CRYSTAL IOS

The role of the several domain ontology work packages within the CRYSTAL project was discussed together with SP6 partners in Munich 26th of November 2013 meeting. Below we summarized the results and agreements of this discussion.

- IOS concerns are defined following up the existing OSLC domains.
- OSLC identifies domains of tools in which resources of different types are used in combination with each other to serve identified activities in the lifecycle of products. Tools in each of these domains maintain closely connected resources. OSLC working groups were established to provide specifications for exposing server interfaces to manage these resources and establish links between them. Example domains are Requirement management (OSLC/RM), Architecture management (OSLC/AM), Change management (OSLC/CM) and so forth.
- Each CRYSTAL domain ontology work package will look into which concepts play a role in the engineering cycle of the specific domain/use cases. As a starting point for the

development of the ontology, a concern from the list of IOS concerns can be chosen (for example: architecture management, risk management). The first iteration in the ontology development process will in this way be an ontology that only supports this concern. In later iterations it can be extended towards a more complete ontology that is based on and contributing to more, and potentially all, steps of the engineering cycle.

- A consolidation activity of the concurrent concepts out of the different domains will be organized in the context of SP6.

The exercise described above will result in extensions for the core vocabulary of IOS/OSLC.

There are different approaches for performing extensions, having the objective to provide:

- New types/concepts
- New properties/attributes
- New links/references
- New specifications (think of: safety management, variability, etc.)

2.3 Scope of the domain Ontology

The ontology coverage could spread from requirements definition toward system design, manufacturing and in-service maintenance. Yet in the present context, we will limit it to the description of concepts and objects used in the collaborative design activities of the SP2 Aerospace domain use cases, and especially in the Engineering Methods (EM) to be applied to these use cases. A review of the Engineering Methods addressed in CRYSTAL public use case is detailed in deliverable [

|Kyo, 1990]	Kyo C. Kang and al. Feature-Oriented Domain Analysis (FODA), 1990.
IWSSD '89	Proceedings of the 5th international workshop on Software specification and design 152-159. 1989
[Moros, 2008]	Begoña Moros, Cristina Vicente-Chicote, Ambrosio Toval, *Metamodeling Variability to Enable Requirements Reuse, in* Proceedings of EMMSAD 2008
Siegmund,	Norbert Siegmund, Martin Kuhlemann, Marko Rosenmuller, Christian Kaestner, and Gunter Saake, Integrated Product Line Model for Semi-Automated Product Derivation Using Non-Functional Properties, in
CRYSTAL D208.010	CRYSTAL aerospace use case description Report – V1 D208.010, 2014.

]. They are:

- Analyze Requirement
- Verify Design against Requirements
- Maintain consistency between multi-viewpoint models
- Trade-off Analysis
- Generate Fault-trees
- Heterogeneous Simulation
- Provide specification document
- Provide Process Management
- Change Impact analysis
- Traceability Matrix
- Search Data

For example, for the needs of UC 2.2 that needs identifying which information is exchanged between designers and PLM, the ontology shall provide unambiguous definitions for concepts related to product development activities, engineering processes specification (e.g. phases, activities and work products).

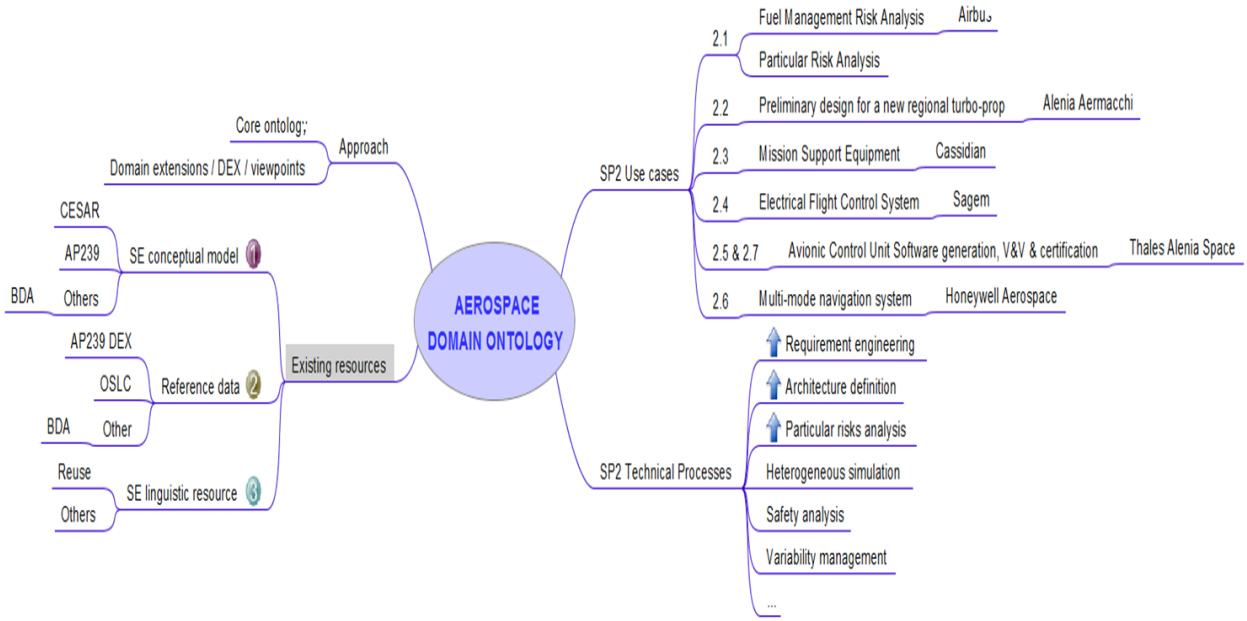


Figure 2-2: Preliminary vision of the Domain Ontology context

2.3.1 Ontology coverage in synthesis

The schema here below proposes a possible structuring of the envisioned coverage of the domain ontology. It may lead to further specification of several interlinked ontologies.

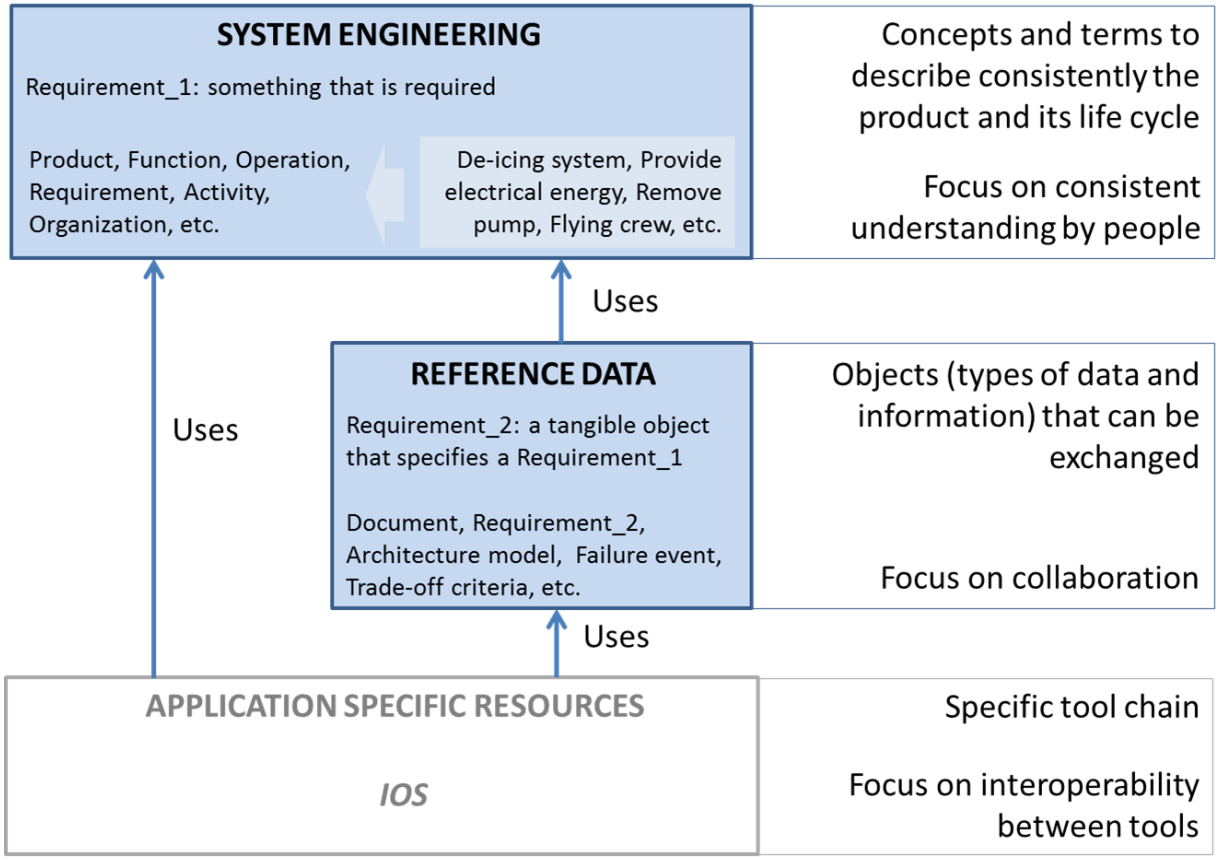


Figure 2-3: Layered scope for a domain ontology

2.3.2 SE concepts in aerospace domain

The domain ontology can provide a description of the System Engineering concepts used in the aerospace domain. This includes the usual SE high level concepts such as:

> *Typical concepts: Product, Function, Capability, Operation, Requirement, Activity, Organization, Scenario, etc.*

But this may also include **refinement** of the concepts as required for the description of the domain related products and products life cycle.

For example terms and definitions for Flight Phases are domain specific. Domain actors (Flight crew, ATC, Passenger, Maintainer, etc.), disciplines typically involved in an A/C development

(safety, operations, installation, human factors, etc.), A/C typical components (fuselage, wings, landing gears, interiors, etc.) or functions (Provide thrust, Resist to differential pressure, etc.) are other possible examples of Domain specific concepts that must be named and identified with no ambiguity.

The approach to identify the high level concepts is rather top-down: the intent is to set up a framework to be used for interoperability at a conceptual level. The ontology shall present the essential information in a way that is understandable by the many stakeholder communities involved in SE activities. It shall ensure a shared understanding of the activities, assets, methods, objects used when performing SE.

As part of the state of the art in this area, the Product Life Cycle Support STEP standard (**PLCS/AP239)**, the **VIVACE Virtual aircraft model** and to some extend the **CESAR conceptual meta-model** in the CESAR project's modelling approach are valuable inputs to be considered and reused. These existing inputs are further described in this document.

2.3.3 Reference data & information shared in SE process

The interoperability among tools and services that are part of the System Engineering environment would benefit from the adoption of standardized and richer data definition models. In this perspective, the ontology can provide a **description of the types of exchangeable data/information used when performing SE technical processes**. It would define the abstract properties of classes of data objects.

> *Typical exchangeable objects: Requirement, Requirement collection, Architecture model, CAD Model, Parameter, Parameter value, Fault tree, MMEL condition collection, etc.*

This time, the approach to identify the reference data is rather bottom-up and pragmatic. The description is done from the business standpoint first, in order to give meaning to digital information (data, files) that must be retrieved or exchanged. It must be such that any kind of domain operational data or information can be mapped on it. We have to look at the Engineering Methods defined for the use cases, which constitute as many information exchanged scenarios (to understand usage scenarios first).

The link with the upper layer is straightforward: the SE concepts can actually be used to express the context for the production or usage of the operational data.

The reference data model is intended to support a capability to exchange information: one or many information services will be defined using this model.

As part of the state of the art in this area, the methodology is rather comparable to the DEX (Data Exchange Specification) for ISO 10303-239 PLCS. (See below § 5.1.2). Previous research projects like CRESCENDO ended with proposals for characterizing exchanged data and artefacts as a DEX. Simulation data exchange is a sub-case.

The approach is also in line with the IOS objective and technical choices in the project: OSLC types of exchangeable “resources” for various Concerns. The list of IOS concerns includes for example Architecture management, Risk management, etc. (see Annex II). The IOS taskforce (to be set-up) will be concerned with the consolidation of the concurrent concepts out of the different domains in CRYSTAL, and this will result in extensions for the core vocabulary of IOS/OSLC.

2.3.4 Specific application resources

Any foreseen Engineering Method should be possibly described using the above mentioned concepts. But UCs implementations will be done using given tool chains. In this perspective, the ontology may support the definition of **application specific resource** to directly support some demonstration, notably in the public use case.

This may require a lower granularity level of the information to enable identifying relevant instances of reference data.

Beyond the scope of the ontology, SP6 services will be developed that basically allow getting or pushing information from and towards Engineering Tools. This requires that the above mentioned concepts are indeed relevant together with the languages and formalisms involved in the UC tool chain. A refinement or extension of the reference data model may be required. If formalisms do not have any explicit abstraction levels, making the mapping in concrete cases may be model dependent.

3 Partner SoA regarding SE concepts and reference data modeling

This section provides a synthesis of the State of Art reached by partners regarding SE concept and reference data modeling. It includes both the knowledge obtained from previous research initiatives and the concepts that are applied in daily activity for system engineering.

3.1 Industrial Partners SoA

With the support of different internal and research initiatives that also involved the participation of National research institutes and Academic bodies, AleniaAermacchi investigated different aspects of “System Engineering” related information structure and relations.

One of the main focuses has been on identifying proper data-models (ontologies) for representing System “Functional View” through the different life cycle phases, starting from “As required” product toward the “As designed” product.

The “as required” view represents the product’s requirements configuration as foreseen by the “contract” that has been defined with the customer, emerging from market surveys and from applicable standard and processes.

The “as conceived” view has the objective to support the product’s concept definition phase by handling the specification of the different envisaged technical solutions, together with their trade-off analysis. It allows to trace the acquired know-how and innovation items storing them for further studies at domain level.

This effort produced the definition of a “System Functional View” in terms of a data model that covers the structure and relations of requirements, functions and the logical architecture.

This vision, which is graphically represented in the following Figure 3-1, is considered a basis for contributing to a common understanding at domain level for System representation information.

It may also help in consolidating a common understanding about a “Traceability View” for the different artifacts that have been produced during the product development life cycle.

It obviously needs to be harmonized with the emerging information models in this context, including AP233 as referenced in section 4.

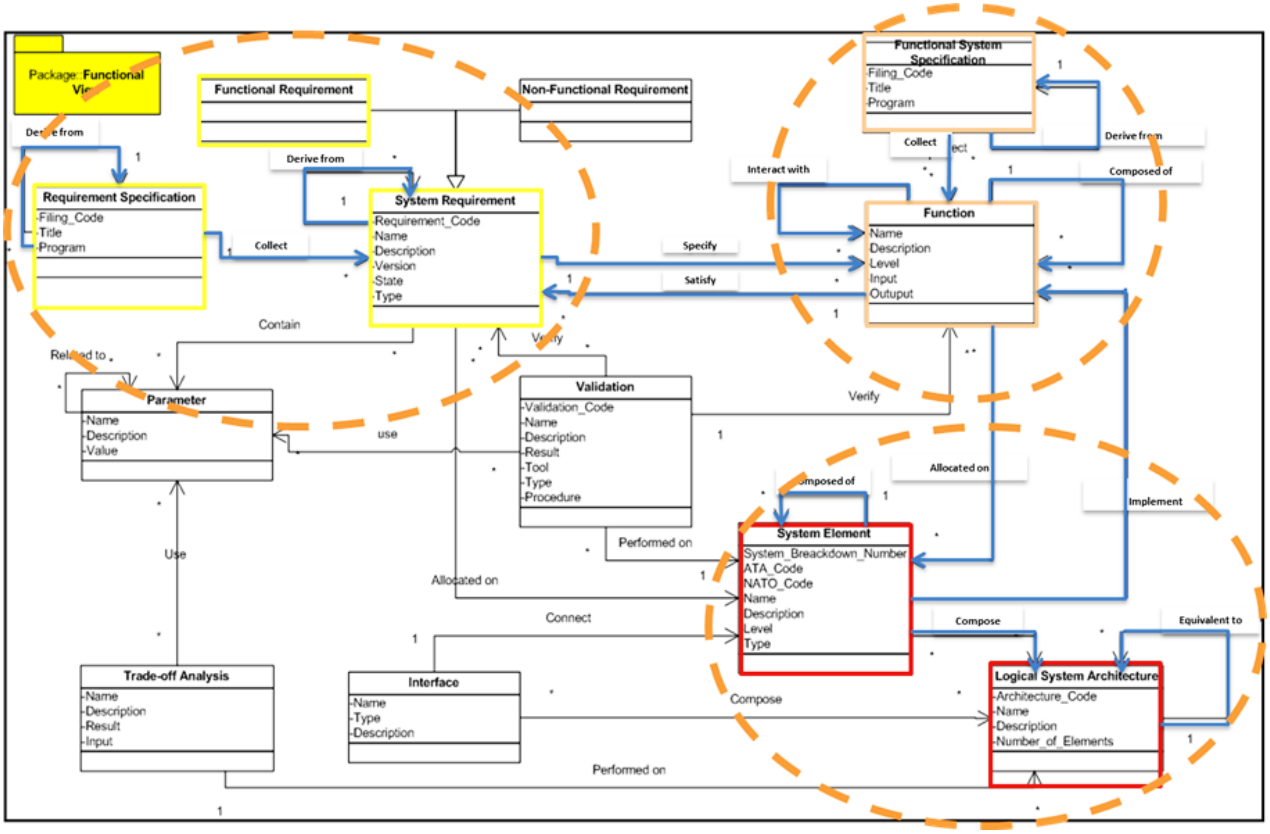


Figure 3-1: Conceptual model of the Functional view

This conceptual data model identifies concepts and relations related to requirements, functions and System elements. These concepts and their relations may assume different meanings according to the context or domain of application. In order to clarify our vision, a detailed explanation is provided in a dedicated table in Annex IV to this document.

Concerning its involvement in supporting the traceability view, here below the approach is graphically represented.

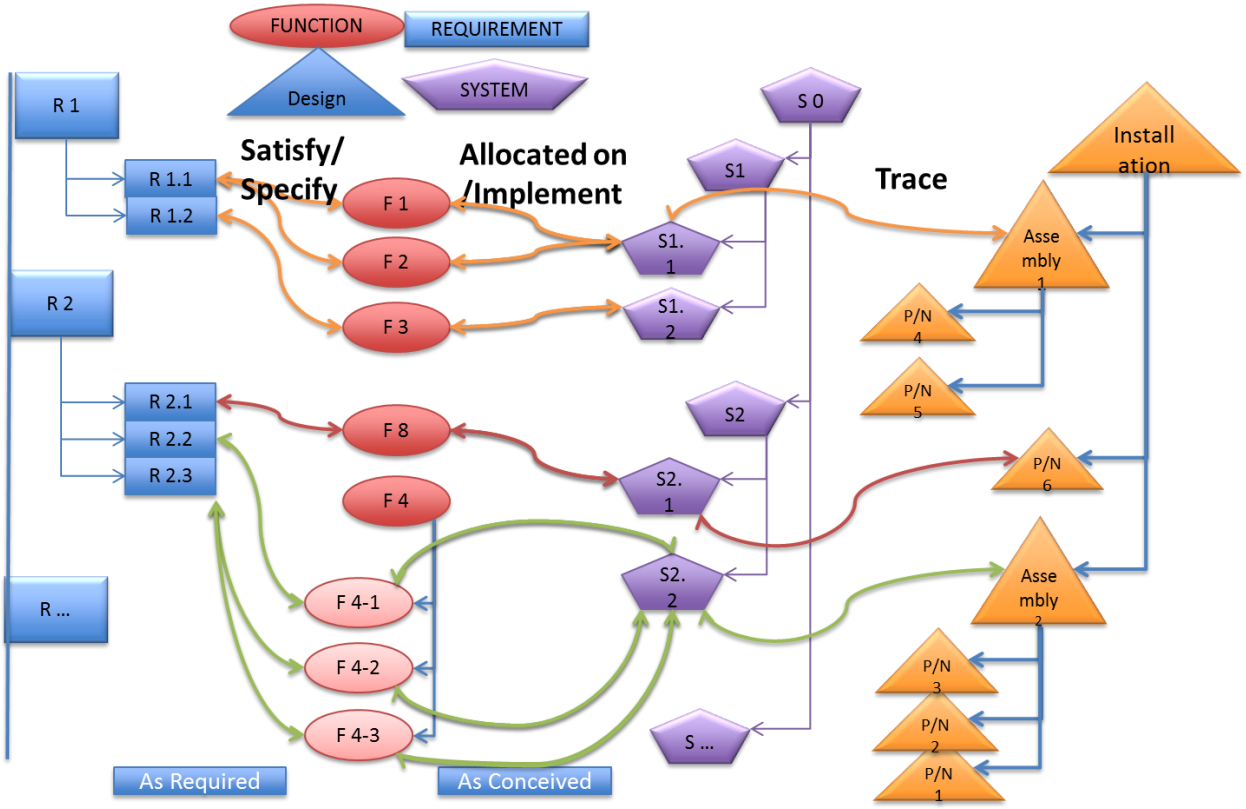


Figure 3-2: Approach for the Traceability View

Airbus and Innovation Works contributed to define frameworks for the Engineering Data Management of the Virtual Aircraft (VIVACE project) and for collaborative modelling and simulation for A/C development (CRESCENDO project, http://www.crescendo-fp7.eu). The most recent output of these past works is nowadays represented by the Behavioral Digital Aircraft (BDA) model developed under the CRESCENDO project.

In VIVACE, a conceptual model was built to describe a generic process for A/C system development [REF D2.1.1.2]. The **Virtual Aircraft model** provides reusable concept definitions of:

- A/C life-cycle stages
- life-cycle processes
- typical breakdown for A/C system
- typical breakdown for A/C functions
- operating phases

as well as some typical simulation scenarios. The Virtual Aircraft model description uses UML-like notations (Figure 3-3).

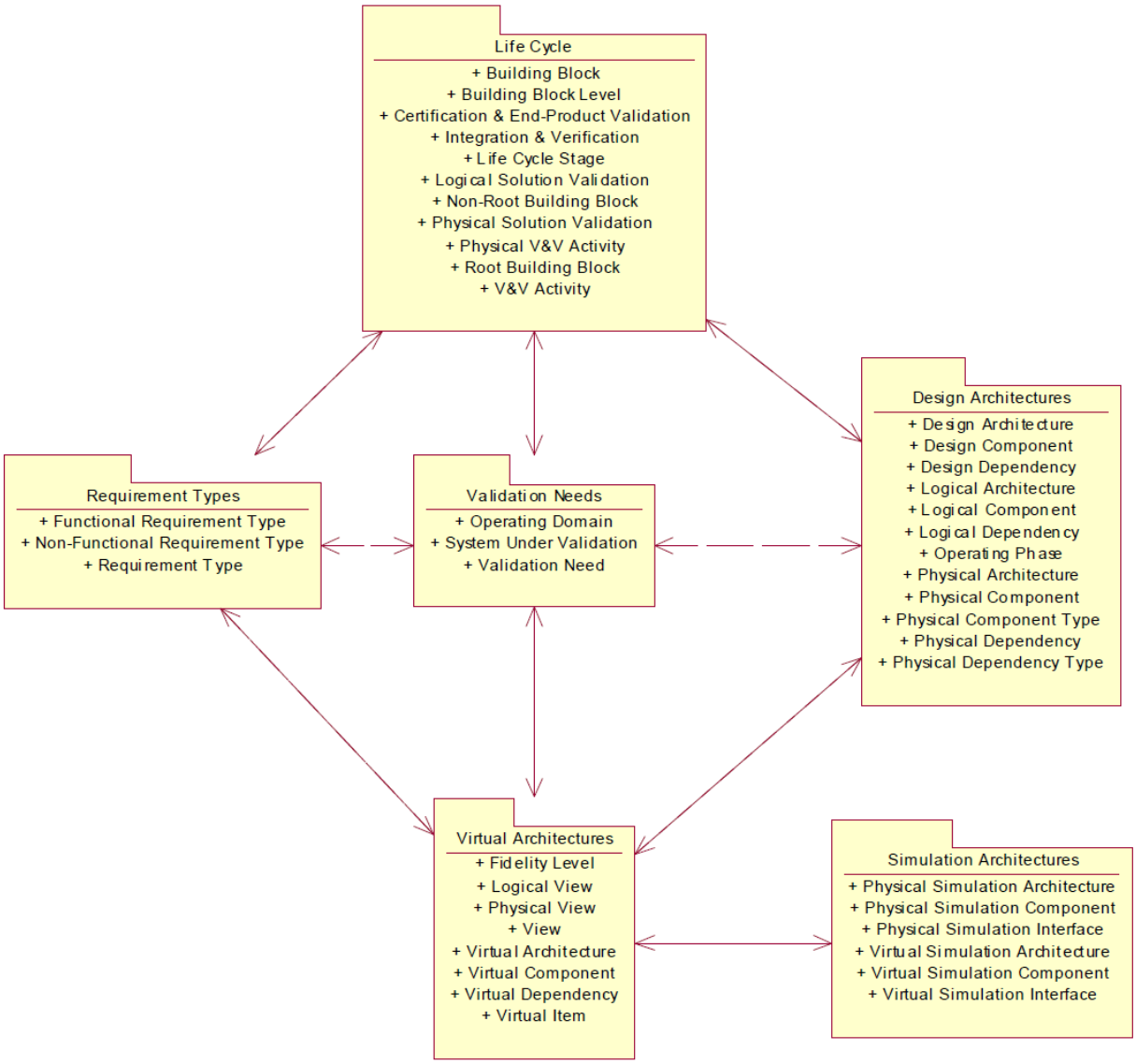


Figure 3-3: Virtual aircraft model overview

The VIVACE Engineering Data Management (EDM) framework included three main components: the Domain Model, the Information Model and the Consolidated Model or Repository (Figure 3-4).

- A "Domain Model" describes the subset of a business user's ontology. It is a schema of the concepts and their inter-links used to perform an activity. The domains users (mechanical, aerodynamic, thermal, system …) from the different engineering disciplines (design, simulation, test …) describe a subset of their model. By this, they define their view on the

product and delimit the information they want to share for collaborative engineering activities.

- The Information Model role is to provide the references between the various product representations (CAD-Models, simulation data, test data ...) and the context (project targets, derived requirements and constraints, process, decisions, knowledge, development phase and lifecycle) they are created or used in. Context description elements are provided by the Virtual aircraft model.
- The several Domain Models can be stored and merged in the Consolidated Repository.

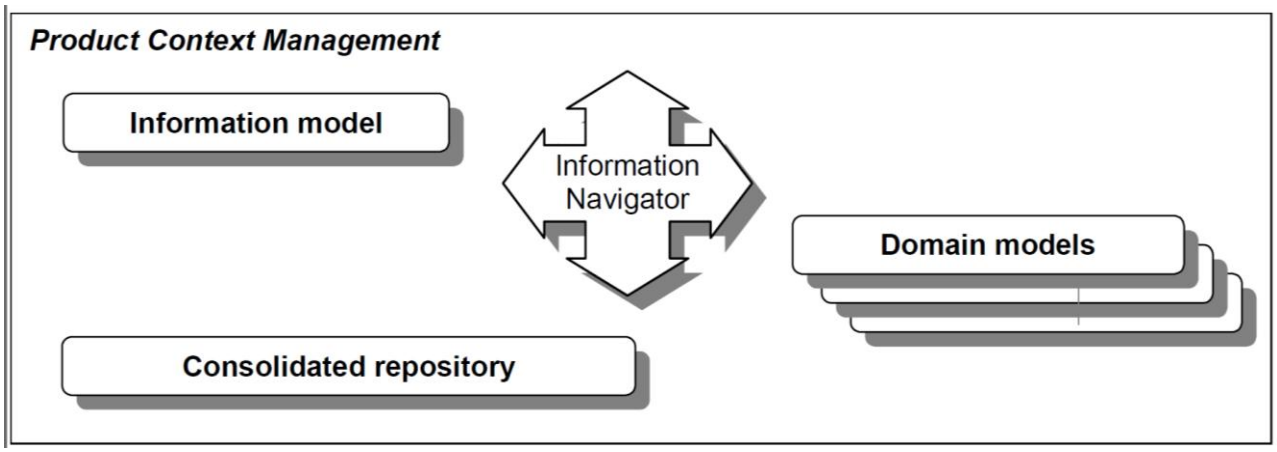


Figure 3-4: EDM main concepts

The VIVACE framework implementation used standards XML and STEP application protocol 239 (PLCS) (see below §5.1.1). Mechanisms to describe and to exchange the information include the definition of EXPRESS schema, their conversion to XMI (UML), the import and further edition (extension, stereotype) of these UML models in some UML modeling environment, and generation of a platform specific application called the Information Model Manager (IMM) from the created UML models. The IMM then offers services such as navigation, search, CRUD or import/export.

In CRESCENDO project, similar HUB collaboration principles were applied within a different infrastructure, and the models were a step further detailed. The Behavioral Digital Aircraft (BDA) is an enabler for conducting collaborative modelling and simulation amongst distributed partners. The BDA model especially introduces and describes fundamental concepts for collaborative M&S activities such as Study, Associative Model Network, Model Instance, Key Value Instance, as well

as Methodology Objects (Collaborative Model Template, Model Type, and Key Value Type) and Object to represent Requirements and Verification and Validation.

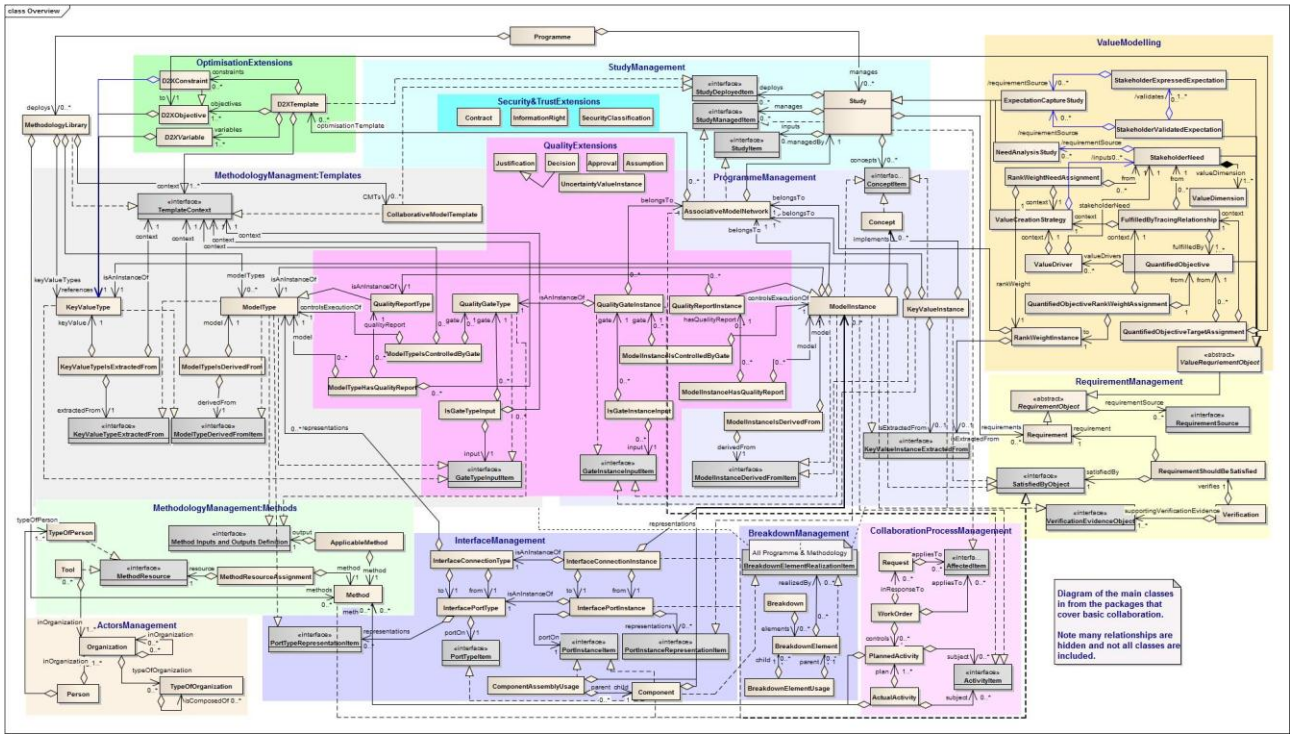


Figure 3-5 – BDA model overview

Regarding the framework, the collaboration model is based on the contract drawn up between the participating companies, and implemented by the Collaboration Hub technology deployed in the project. In CRESCENDO, Eurostep Share-A-space 7.3 operated as the BDA collaboration hub: it provided the repository for sharing SLM and PLM information and providing the business services needed to manage the information shared in the hub. It hosted the BDA web-services based on the BDA business object model.

The BDA is now utilized and continued in the context of new ongoing research initiatives. Both TOICA and CONGA projects are developing tools and methods for the architects to conduct and monitor the trade Studies, each focusing on different aspects, but sharing a common set of functionalities, such as requirements management and study creation and orchestration. TOICA especially is seen as a direct continuation of CRESCENDO for a more limited scope of Thermal aircraft M&S.

Both TOICA and CONGA are also working towards the standardization of the BDA Business Object Model under the MoSSEC project. “The MoSSEC project is designed to provide a capability to share modelling and simulation (M&S) information in a collaborative systems engineering context, and to enable full traceability and re-use of this information throughout the product lifecycle and independent of the specific IT applications used across collaborating enterprises.

The standard covers a core subset of systems engineering information content and the related practices and information services necessary to support engineering collaboration.

The MoSSEC international project brings together the works started in Europe by CRESCENDO Consortium to apply AP239/PLCS for this scope, and similar works in Europe and US motivated by similar goals. It is anticipated that the form of the project deliverable will be a DEX that will use a subset of the AP239/AP242 harmonized information model, and a set of related web-services that efficiently support the operational practices of the DEX, including both atomic and aggregated information operations.” [MoSSEC 2013].

These ongoing initiatives represent a big effort that clearly must be acknowledged in CRYSTAL Aerospace domain ontology definition in order to avoid any divergence detrimental to the standardization objective.

3.2 Ontology for requirement controlled authoring

The REUSE Company brings in the CRYSTAL project a technology to address issues related to requirement authoring and quality assessment. Requirements Quality Suite (RQS) is a commercial tool aimed to customize, manage and improve the quality of a set of requirements. RQS includes a linguistic ontology that describes and structures the possible lexical units that can realize a given variable (placeholder) in a requirement pattern (or boilerplate).

This resource is structured according terminological relationships that convey normative categories such as Broader Term, Narrower Term, Related Terms, Scope notes, etc., as shown in the figure below which illustrates the KM Thesaurus management HMI. It can be multilingual and conveys normalization attributes (Preferred English label, preferred French label).

This resource is utilized in order to build a semantic formalization of a requirement as a semantic graph, which can match a requirement pattern formalized the same way. This allows various

applications for requirement authoring support and requirement quality measurement computation. [CRYSTAL-D607-041]

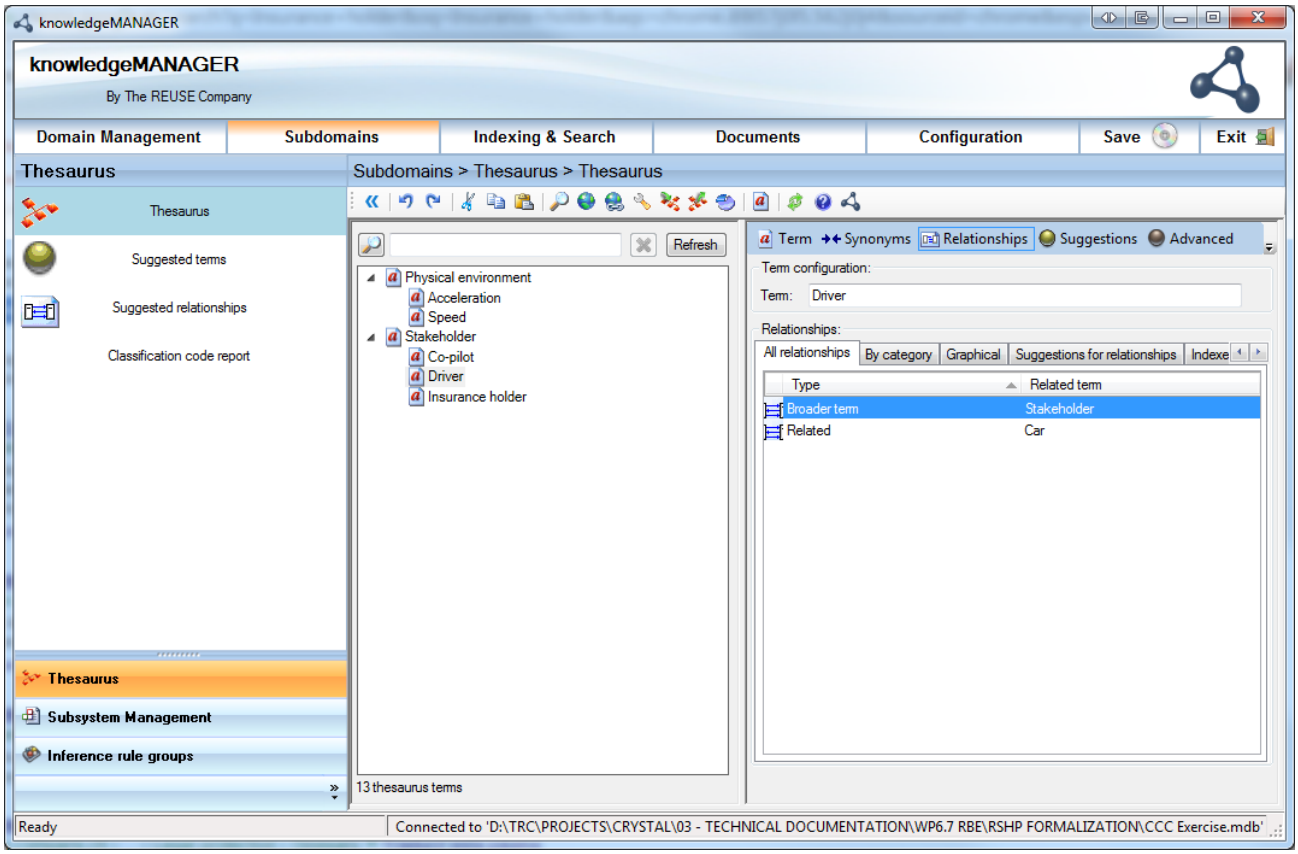


Figure 3-6: REUSE KM Thesaurus management

The RQS technology is part of the SoA in CRYSTAL and various connections with SE and Reference data ontologies have been identified through discussion with WP6.7 partners.

- Firstly, the current indexing and retrieval mechanisms in RQS that are currently applied to the requirement statements could be apply to other kinds of assets. These assets, provided they use the same lexical realizations (known vocabulary) for their description and can be formalized as “match-able” sematic graphs, would therefore be indexed and retrieved (these technologies could be tested with SP6 partners).
- Secondly, the RQS KM ontology resource could be used for the normalization of the names given to concepts, since a term entry in the KM ontology can be described with several labels (preferred, alternative, etc.). We should keep in mind this capability. The coupling with the domain ontology may allow the definition of new services such as the normalization

of the terms associated to the concepts. As a first step, a publication of the KM ontology could be envisaged and render accessible to the several actors implicated in a use case. Figure 3-7 below shows a WP6.7 proposal for representing the data in a SKOS format, here edited in Protégé editor. Various possible lexical realizations for the entry "*pedal*" are shown as annotations (e.g. rdfs:label, altLabel, prefLabel): "*Pedal of the Breaking subsystem*" or "*Pedal component*". These possible nominal phrases that requirements authors might use actually refer to the same "*Pedal"* concept and the preferred English label "*Pedal of the breaking system*" can be substitute to them. One can imagine a similar mechanism to be applied to an architecture Physical Block name for example. A SKOS export of the KB ontology resource is planned at short terms by WP6.7.

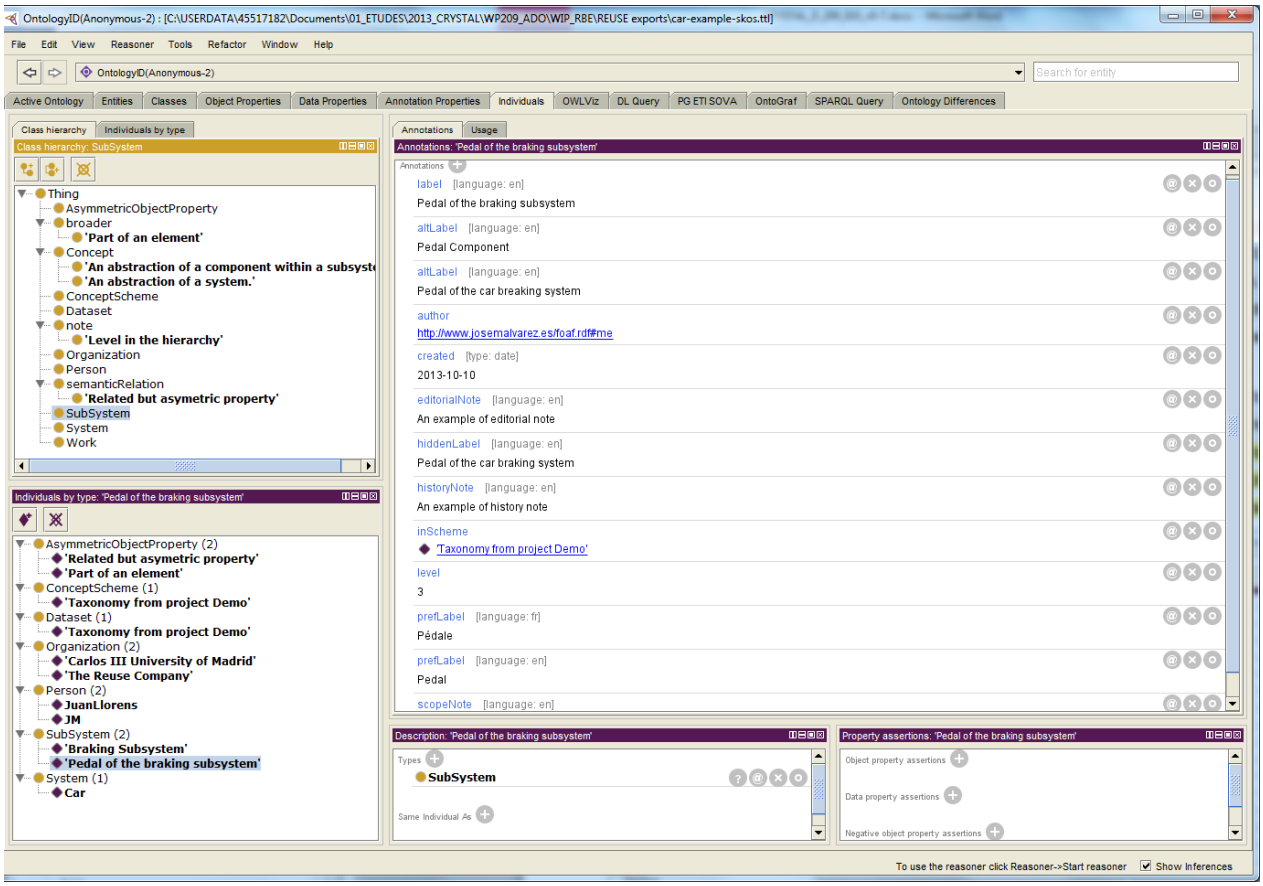


Figure 3-7: SKOS representation of RQS ontology

Finally, it is foreseen that the REUSE Company technologies will evolve during the CRYSTAL project duration towards more advance parametrized requirement patterns (see

Figure 3-8 below borrowed from WP6.7 “from IOS Services from REUSE Company for RBE” presentation). This aspect impacts the domain ontologies definition since description of parameters names, physical dimensions, quantities and units are needed. It is further discussed in section § 6 below.

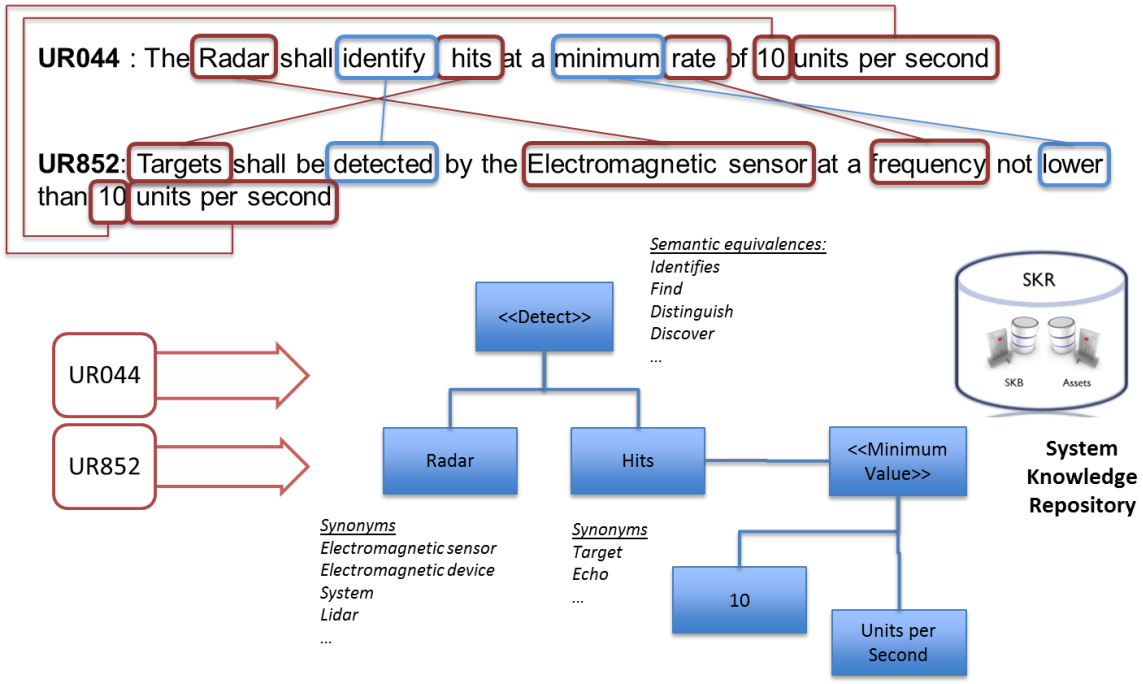


Figure 3-8: Representation Schemas

4 Standard for Engineering Data representation

This section provides a synthesis about standards and techniques currently available for the formal representation and exchange of engineering information. This synthesis is based on the know how gained by CRYSTAL partners through the participation in previous research initiatives and through their internal initiatives. In this respect it shall be considered that, when dealing with the analysis and design tool-chains, we detect a specific demand from the tool providers and IT specialists for the adoption of clear and standardized data-models.

Here below we are mainly focusing on the AP233 protocol that is being considered for a multi-domain adoption and not limited to the Aerospace context.

The knowledge developed in the framework of the ARTEMIS-CESAR project with respect to the management of “Viewpoints” at System level is also reported.

4.1 Overview

According to [**INCOSE 2020**] vision, MBSE is the formalized application of modeling to support system requirements, design, analysis, verification and validation beginning in the conceptual design phase and continuing throughout development and later life cycle phases.

It aims at improved quality in development, increased productivity and risk reduction. Given this scope, and the wide range of tools used to assist the Systems Engineer in performing their tasks there is a clear and obvious need to enable data exchange between these tools.

AP233 is the ISO Standard **Data Exchange** Protocol that provides this support and allows the Systems Engineer concentrate on their tasks instead of regularly dealing with data exchange problems. It enables:

- communication between similar tools in Systems engineering usage
- communication between different tools both within and external to Systems engineering functions
- holistic approaches to data sharing
- fine grain configuration management to Systems engineering data
- sharing of management data currently locked up in proprietary formats and tools

It also Leverages Other Standards. It links to 30+ other ISO STEP information model standards (called Application Protocols), including:

- Configuration Controlled Design (AP203)
- Engineering Analysis (AP209 E2)
- Produce Lifecycle Support (AP239/PLCS)

AP233 and OMG SysML are aligned and were developed by the same INCOSE team (MDSD) and share many concepts.

Here below the overlaps between AP233 and the Product Lifecycle Support information model are shown through a graphical representation.

AP 233, *Systems engineering data representation*
AP 239, *Product life cycle support*

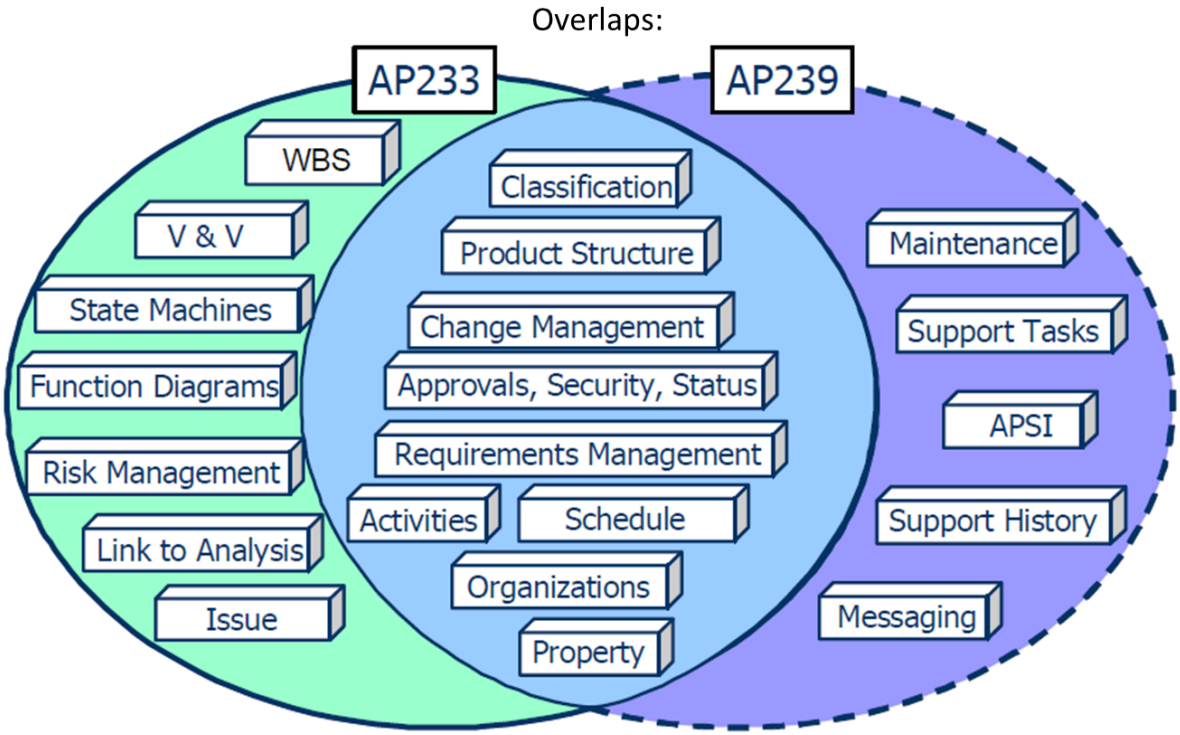


Figure 4-1: AP233 vs. AP239 overlaps

AP233 Extensibility

By use of generic structures and data elements AP233 does not constrain its usage. Specific usage and semantics are added using external class libraries that can be viewed as “Reference Data Libraries” or a sort of “Taxonomies”.

The semantic extensions can be specified in OWL (Web Ontology Language). Some are standardized within AP233 itself.

4.2 Requirements data

AP233 provides support for Requirements Management in the same way that is supports the management of other kinds of items, by treating a Requirement as a kind of **Product**. This enables Requirements to have versions, different definitions and different representations, and be managed by organizations, with associated dates, status, etc.

The following UML diagram shows the basic structure of the AP233 Requirements concepts.

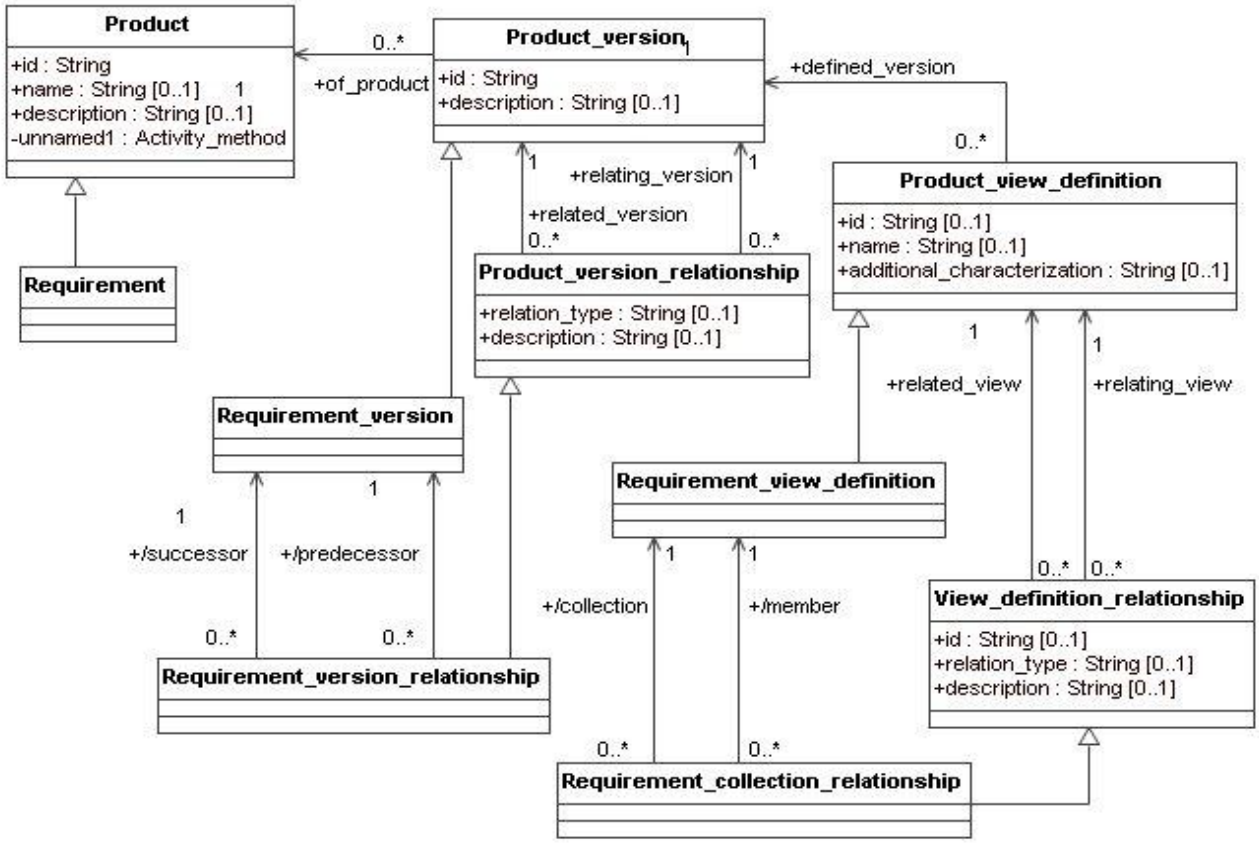


Figure 4-2: AP233 Requirements concept structure

AP233 supports versioning of requirements and allows for multiple definitions of the same version.

4.3 System Breakdown

In AP233, breakdowns are represented as a set of objects that may be change managed and version controlled. The breakdown of systems into constituents can also be change managed and version controlled. This capability is wider than that is provided by most diagram-based tools for system engineering. However, it enables rigorous control of the systems descriptions as they go through their development. The breakdown itself is treated separately from the thing it breaks down.

These concepts are outlined in the following figure.

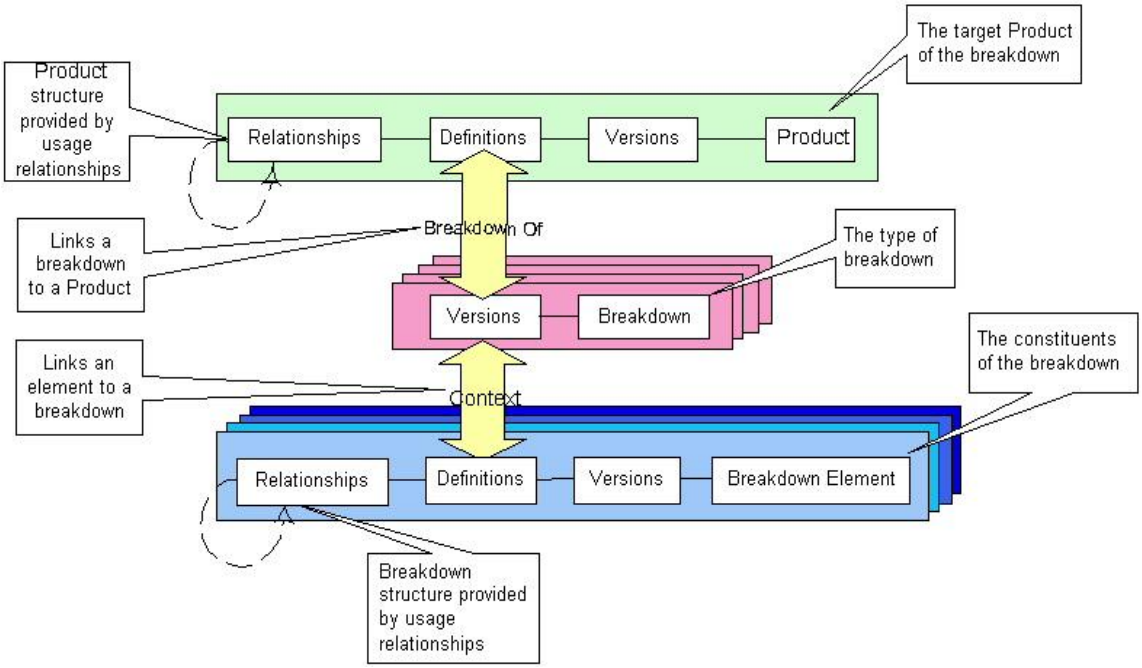


Figure 4-3: Breakdowns concepts

Several types of breakdowns are supported by AP233. These are described as follows.

- System breakdown (e.g. systems and subsystems)
- Functional breakdown (e.g. functions and sub-functions)
- Physical breakdown (e.g. a real-world breakdown)
- Zone breakdown (e.g. based on a spatial location)
- Hybrid breakdown which contain a mixture of the other types of breakdown elements

The following UML diagram provides more detail about how breakdowns work in AP233.

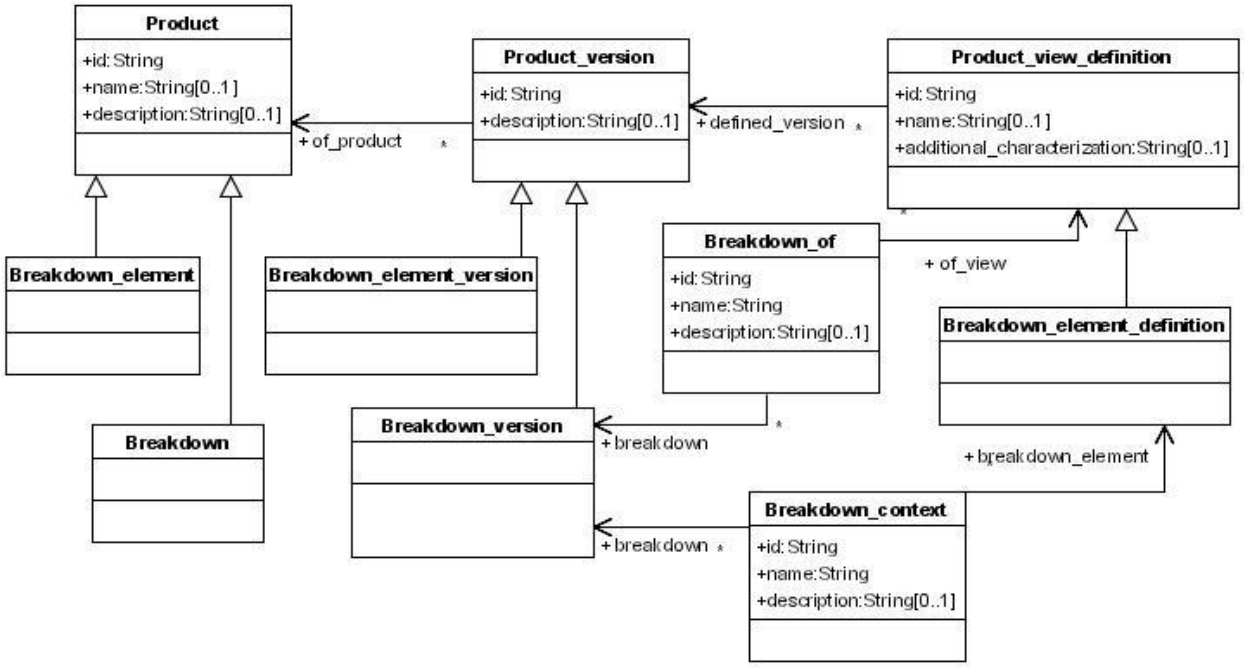


Figure 4-4: Breakdowns structure in AP233

In AP233, **systems** are represented as a set of objects that may be change managed and version controlled. System breakdowns are kinds of breakdown in AP233 (see AP233 breakdowns for an overview).

The elements involved are as follows.

- The link between the thing being broken down and the breakdown itself.
  - System_breakdown with an assigned identifier and child elements Name and optional Description each containing text.
  - System_breakdown_version with an assigned identifier refers to the System_breakdown using the Of_product child element and has an optional child element Description containing text.
- The identification of the elements of the system breakdown.
  - System_element with an assigned identifier and child elements Name and optional Description each containing text.

- System_element_version with an assigned identifier refers to the System_element using the Of_product child element and has an optional child element Description containing text.
- System_element_definition with an assigned identifier and refers to the System_element_version using the Defined_version child element.
- System_element_usage classified as a Decomposition, which defines the decomposition relationship for the system breakdown, with a child element Relating_view referring to the parent System_element_definition in the decomposition and a child element Related_view referring to the child System_element_definition in the decomposition.
- System_breakdown_context with an assigned identifier and refers to the System_breakdown_version using the Breakdown child element and refers to the System_element_definition using the Breakdown_element child element.

4.3.1 Defining viewpoints

There are different approaches to define viewpoints. Here below we are providing a comparison of the approaches adopted by the ISO 10303-239 Information Model with the approach defined by the ARTEMIS-CESAR project.

4.3.1.1 Views as part of PLCS ISO 10303-239 information model

We are referring here the part 1019 of ISO 10303-239, whose definitions are the following:

- Characterization of a version of a product, suitable for one or more application domains and one or more life-cycle stages;
- Identification of a universe of discourse suitable for the description of products;

The “Product view definition” according to the mentioned specification is shown in the following diagram that includes the relations with other concepts.

It helps in clarifying how the “view” is related with the product lifecycle and application domain. An effort should be produced in order to “map” the different existing visions about viewpoints toward an accepted and shared specification such as this one.

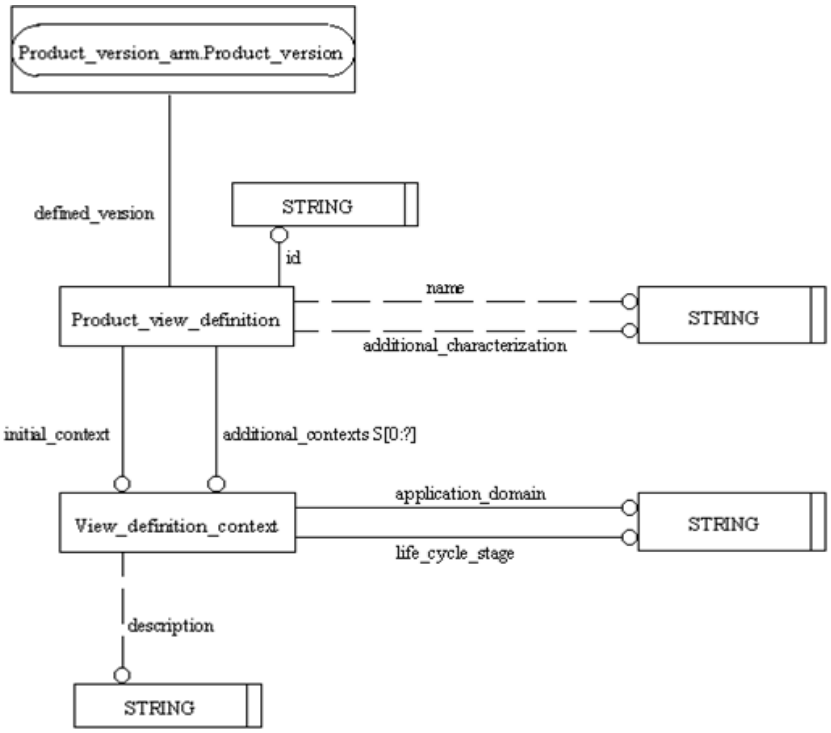


Figure 4-5: View model

4.3.1.2 Viewpoints in CESAR CMM

In the context of CESAR project different viewpoints have been defined by using three abstract categories: Abstraction level, Aspect, and Perspective.

a) Abstraction level: five abstraction levels are defined to reflect the progress of the SE analysis process
   - Operational analysis
   - Functional/non-functional need analysis
   - Logical architecture analysis
   - Physical architecture analysis
   - Implementation level (realization)

b) Aspect: “The concept of aspect classifies the design specifications and realizations by addressing specific *concerns*... A *concern* characterizes a specific interest of one or several stakeholders with respect to the system under development”. Four aspects were proposed:

- Safety: define the dysfunctional aspects of the system
- Real time
- Variability (~Product line: define the variability points)
- Functional behavior

Many other concerns could be envisaged, and some are mentioned e.g. performance, reliability, security, distribution, evolvability, etc.

c) Perspective: A perspective is a way to regroup in a transversal way some concepts pertaining to different abstraction levels

- Operational perspective: focus on the operational aspects of the system
- Functional perspective: focus on the functional aspects of the system
- Logical perspective: define system architecture, define abstract components, allocation of functions on them, behavior of components and interfaces between components
- Technical perspective: define concrete hardware and software components, allocation of functions on hardware and software components
- Geometrical perspective: define spatial dimensions of the components

Each viewpoint is described as an association of three characteristics: the concerns addressing the modeling, the methods to be used.

Core viewpoints concerns

In the following table a set of core viewpoints that were identified and considered suitable for the scope of embedded safety critical systems from CESAR is described.

To identify a first set of viewpoints the issues addressed along a typical engineering cycle have been collected and categorized. The result of this brainstorming among the CESAR's industrial partners is reported in the table. The table provides the main concepts handled in each engineering phase. Each column represents an example of a viewpoint (such as safety or performance) and each line describes the engineering phase and the content that must be included in this viewpoint at each engineering phase.

		VIEWPOINTS			
		System Description		Safety	Performance
		Decomposition	Behaviour		
ENGINEERING PHASES	Operational Analysis	* actors * activities * exchanges	* scenarios between actors * phases, missions * modes & states	* feared events, failure conditions	* operational latency constraints on activities
	Functional & non-Functional Need Analysis	* expected functions * data * data exchanges between functions	* scenarios between system & actors * functional chains * modes & states	* criticality of functional chains, data, & data exchanges * failure & recovery scenarios	* latency on functional chains * processing complexity of functions (or full behaviour description)
	Logical Architecture Analysis	* expected functions * data * data exchanges between functions * added functions * components implementing functions	*scenarios between system components & actors * functional chains allocated to components * modes & states of components	* redundancy paths * vote, monitoring… components behaviour * dysfunctional behavioural model for functions & components	* estimation of function resource consumption * estimation of data and exchanges complexity, rate…

	Physical Architecture Analysis	* expected functions * data * data exchanges between functions * added functions * components implementing functions * implemented functions (incl. Technical) * behavioural components (e.g. software) implementing functions * implementation components (e.g. computers) delivering resources to behavioural components * interfaces between behavioural components * physical links between implementation components	* scenarios between system components & actors, using interfaces between components * functional chains allocated to components * modes & states of components	* redundancy paths * vote, monitoring… components behaviour * dysfunctional behavioural model for functions & components * component failure scenarios	* estimation of function resource consumption * estimation of data and exchanges complexity, rate… * resource available for each implementation component * bandwidth available for each physical link * resulting resource consumption of functions, behavioural components...

Table 1 - Main concerns handled by viewpoints

4.4 Product Structure and configuration

AP233, as are all ISO 10303 Application Protocols, is a Product-centric information model. Other information about concepts such as Organization or Activity also exist but less capability is defined in support of those concepts.

The following indented list shows the hierarchy of kinds of Product, Product Version and Product View Definitions currently defined in AP233.

Product
    Attachment_slot

Breakdown
- Functional_breakdown
- Hybrid_breakdown
- Physical_breakdown
- System_breakdown
- Zone_breakdown

Breakdown_element
- Functional_element
- Physical_element
- System_element
- Zone_element

Document

Interface_connector

Interface_specification

Part

Product_as_individual

Requirement

Product_version

Attachment_slot_version
- Attachment_slot_as_planned
- Attachment_slot_as_realized
- Attachment_slot_design

Breakdown_element_version
- Functional_element_version
- Physical_element_version
- System_element_version
- Zone_element_version

Breakdown_version
- Functional_breakdown_version
- Hybrid_breakdown_version
- Physical_breakdown_version
- System_breakdown_version
- Zone_breakdown_version

Document_version

Interface_connector_version
- Interface_connector_as_planned
- Interface_connector_as_realized
- Interface_connector_design

Interface_specification_version

Part_version

Product_as_individual_version
- Product_as_planned
- Product_as_realized

Requirement_version

Product_view_definition

- Attachment_slot_definition
- Breakdown_element_definition
  - Functional_element_definition
  - Physical_element_definition
  - System_element_definition
  - Zone_element_definition
- Document_definition
  - Digital_document_definition
  - Physical_document_definition
- Interface_connector_definition
- Interface_specification_definition
- Part_view_definition
- Product_as_individual_view
- Requirement_view_definition

5 Standard for Engineering development process

This section provides a synthesis about standards and techniques currently available for the formal representation of engineering processes. This synthesis is also based on the know how gained through the participation to research initiatives and company internal initiatives.

Here below we are mainly focusing on the AP239/PLCS protocol, that is being considered for a multi-domain adoption and not limited to the Aerospace context.

5.1 Product lifecycle support

5.1.1 The PLCS Initiative

Product Life Cycle Support (PLCS) can be defined as “a joint industry and government initiative to accelerate development of new standards for product support information. It is an international project to produce an approved ISO standard within 4 years.

- Commenced November 1999
- PLCS, Inc. closed down 2004
- ISO 10303-239 published in 2005
- ISO 10303-239-CD edition 2, 2009

PLCS plans to ensure that support information is aligned to the evolving product definition over the entire life cycle. It extends ISO 10303 STEP -the STandardfor Exchange of Product model data.

It is seeking to provide a mechanism to maintain the information needed to support complex assets such as ships, aircraft or engines, in line with the changing product over its full life cycle.

It provides extension to the capabilities of AP203 (Configuration Controlled Design) and hence the PDM Schema/Modules, to address the requirements for Configuration Management over the full product life. It will also address the information requirements needed to define and deliver life cycle support for complex assets.

Within PLCS, reference data is used mainly to define standard “types” or “classes” of things, such as classes of documents, types of task, fault codes etc. Reference data is standardized computer interpretable data, which can be used to extend or tailor a data model, or to hold agreed descriptions of data to be shared between participants.

Many current standards could be viewed as potential sources of reference data.

PLCS will use reference data to tailor and extend the AP239 information model in a controlled manner.

Reference data also provides a means for recording common descriptions for any types or classes (e.g. agreed categories of a product, agreed fault codes).

PLCS has based its approach to reference data on that adopted by ISO 15926 (Integration of life-cycle data for process plants including oil and gas production facilities). Further information can be found at http://www.iso15926.org/.

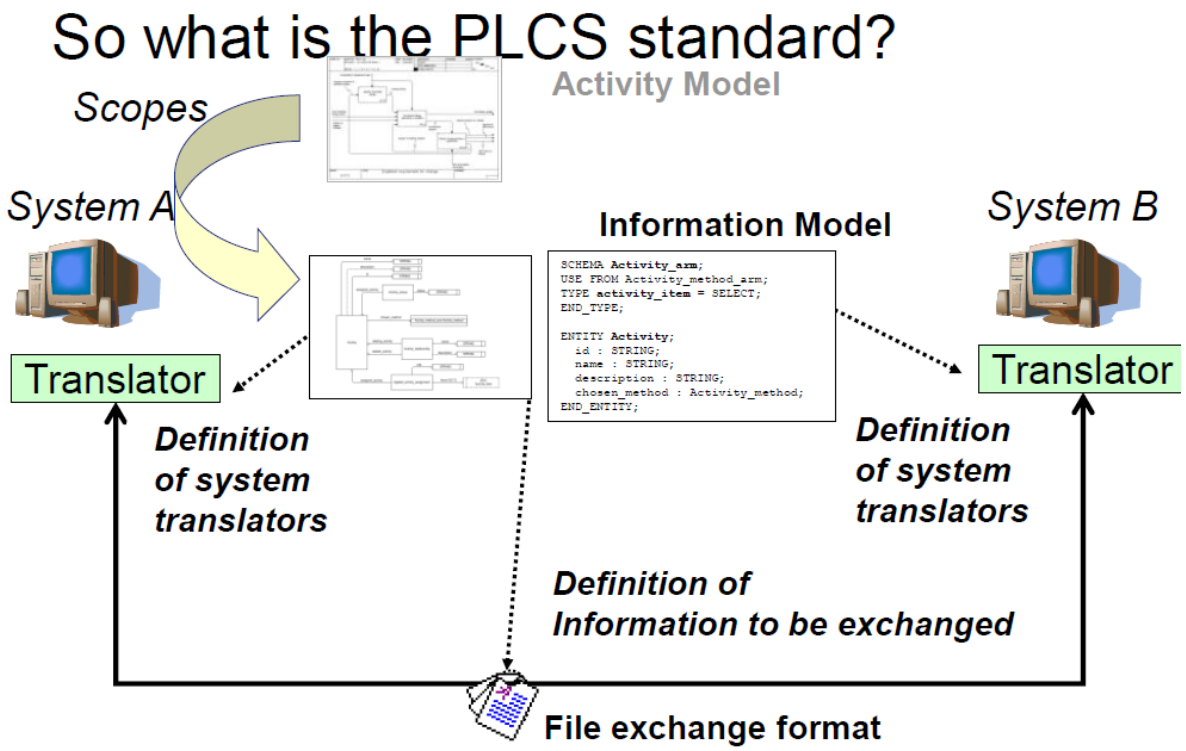


Figure 5-1: PLCS Standard definition according to “Eurostep”

It enables the following capabilities:

Figure 5-2: AP239 Capabilities

Product Description: Capability to define product requirements and configuration, including relationships between parts and assemblies in multiple product structures (as-designed, as-built, as-maintained).

Work Management: Capability to request, define, justify, approve, schedule and capture feedback on work (activities) and related resources.

Property, State and Behavior: Capability that describes and captures feedback on product properties, operating states, behaviour and usage.

Support Solution and Environment: Capability to define the necessary support for a given set of products in a specified environment and to define support opportunity, facilities, personnel and organizations.

A high level concept model for PLCS is represented in the following schema that has been defined and is a copyright of the Eurostep Group, and be retrieved from https://lists.oasis-open.org

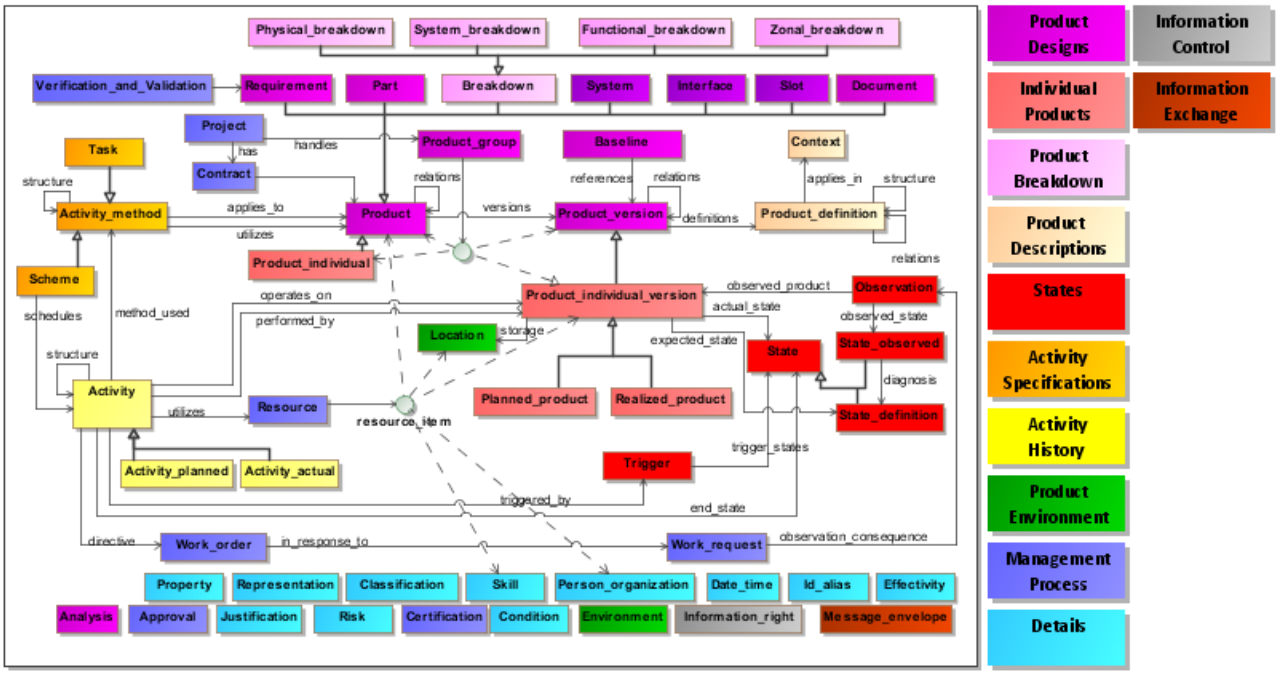


Figure 5-3: PLCS concept model

EXPRESS is the modeling language defined in the frame of the STEP project, and both AP233 and PLCS are primarily formally defined by EXPRESS models.

But formalization for concepts defined by the PLCS capabilities is available in the form of OWL ontology specification language. It is described in Annex III of this document for completeness.

5.1.2 DEX

In the context ISO 10303-239 PLCS standard, different Data Exchange Specifications (DEX) are defined.

A DEX is a subset of the PLCS information model which is defined to support data exchange for specific contexts. A "context" is a context in which a specialized vocabulary is employed. This may be an individual organization, a larger business community, a given business process or a particular project.

Reference data is standardized computer interpretable data, which can be used to extend or tailor a data model, or to hold agreed descriptions of data to be shared between participants.

- DEXs are defined in a separate document from the PLCS Life Cycle Core AP
- DEXs may have additional constraints

- DEXs (as part of PLCS) may specify the use of specific reference data.

The development of DEXs is an ongoing task. A number of DEXs are identified within given contexts and each DEX is defined using the specialized vocabulary of its business context. Examples DEX are (see more complete list in Annex I): Product breakdown for support, Fault states, Maintenance Plan, System Requirements…

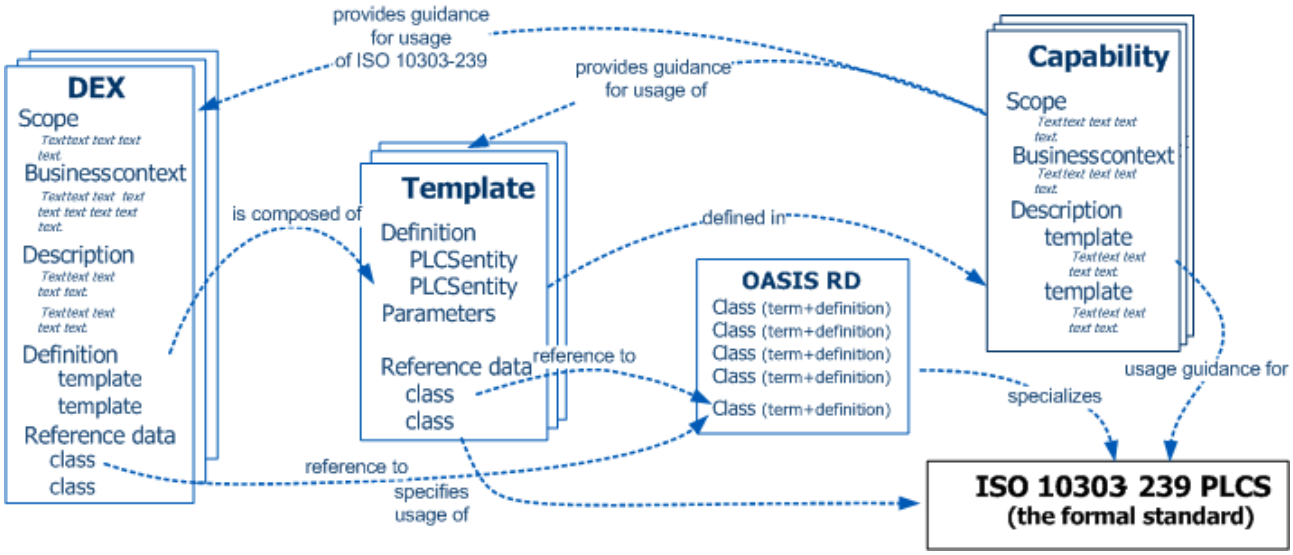


Figure 5-4: OASIS DEX architecture

5.2 Managing Collaboration Processes

According to the knowledge gathered through the CRESCENDO project, the AP239 does not foresee any particular business process. However, a number of constructs are provided for representing the requesting, ordering and recoding of work.

A “Work request” object is used to represent the request for some work to be undertaken. The subject of the request, a part, product as realized etc. is identified. This can result in the initiation of an Engineering Change Proposal (ECP) to investigate various design alternatives. The subject of the ECP, a part, product as realized etc. is identified as inputs to the proposal. Once a number of solutions to the proposal are developed, i.e. new part etc. they are identified as outputs from the proposal. The date when the design is planned to be embodied is represented by a dated planned

effectivity. Having considered the design alternatives, a solution is adopted and the ECP promoted to an Engineering Change Order (ECO).

At any stage of the process a work order can be authorised to undertake further work. The work to be done is represented by “directed activities”. These begin as a planned activity, and then once the work starts, they are started activities, and then completed activities. If appropriate, the activities relate to the tasks used to specify the work to be done and the resource used. Hence a record of all work done is maintained.

The AP239 also provides mechanism for representing plans or schedules – the equivalent of a project plan. This is referred to as a Scheme. A Scheme can have Scheme entries – these are planned activities. They have a planned start date and end date. The scheme entry is assigned to the planned activity which can relate to the task method in order to describe the task to be performed, and to resource items, to indicate the resources that are required to perform the activity.

6 Ontology needs for Requirements Engineering and V&V

6.1 Overview

We reflect here on possible ontology usage and needs to contribute to solving some interoperability issues addressed by the IOS, while remaining aligned with the RBE approach used in CRYSTAL.

Ontologies in the context of requirement engineering may address diverse main needs:

1) how to improve requirement authoring quality through the use of controlled languages, patterns, and diverse NLP technologies. In this area ontologies may address the description of patterns, axiomatic relationships between pattern variables, and lexical resource to be used in connection.
2) how to homogenize requirement management tools and databases through the providing of standardized conceptual models for requirements management which in their turns will allow interchange format definition (INCOSE, OMG ReqIF, etc.).
3) how to ensure continuity of Requirement and V&V Management in MBSE and across the Extended Enterprise.

The requirement authoring aspect is addressed in CRYSTAL through the technical WP6.7 RBE.. Short presentation on the required ontological resource is provided above section §3 as well as considerations regarding possible connections with the here discussed Domain ontology. Other detailed presentations are available in D607.x deliverables

Conventional Engineering methods are based on textual descriptions of the system requirements. A large set of interconnected documents describing a huge volume of requirements expressed in a natural language are produced in the design process. With the introduction of Requirement Based Engineering (RBE), **requirement management models** were defined and implemented in many tools. Requirement specification templates define the editable requirement objects attributes such as title, description, type, url, status, rational, etc. allowing a structured specification of a requirement. Tools and requirement databases such as IBM Rational DOORS and others allow individual requirement management: status monitoring, acces rights, versioning and configuration management, etc. And many consistency checks, data extraction and document generation tasks

can be automated or facilitated. In this area quite stable requirement management models already exist, such as the ones implemented in the CRYSTAL technical partner Requirement management tools; and standards for requirement data exchange exist, such as AP233 *Requirements concept structure* already described above (Figure 4-2).

The latter need, how to enable continuity of Requirement and V&V Management in MBSE, encompasses several CRYSTAL project Use Cases such as Enabling traceability between requirements and models or Supporting design verification against requirements.

6.2 Requirement modeling vs traceability between Requirements and Models

In Model Based System Engineering (MBSE) paradigm, models are used to describe / specify a system. Many modeling languages are actually used, most of them having a graphical notation. UML and SysML especially are representative examples in CRYSTAL. They allow modeling the “intented” architectecture and behavior of the system. A diagram such as a UML state machine diagram (Figure 6-1) is not “completely formal” (in a mathematical sense), but it already “gives a very significant amount of information on the system modes and the rules that drive the related behavior” [Bernard, 2011].

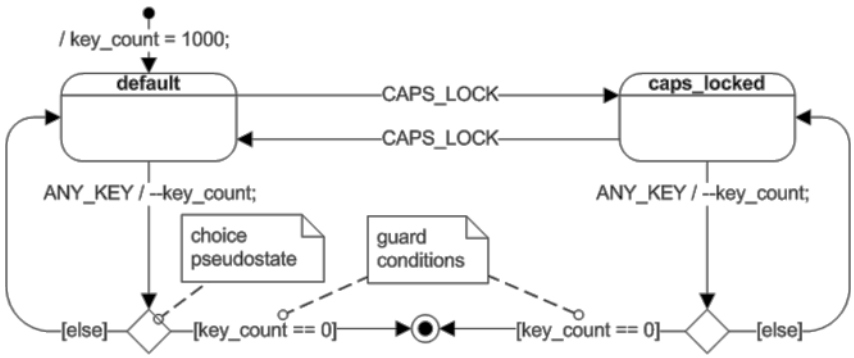


Figure 6-1: UML state machine diagram

These languages are considered by many as means to properly represent the requirements (**requirement modeling**).

Then the produced models themselves shall evolve (and be managed) from their initial “Model of intent” status (representing the intended system behavior) to the “model of specification”, and in

some cases if relevant, the model may even follows a MDA process and be used to generate software code.

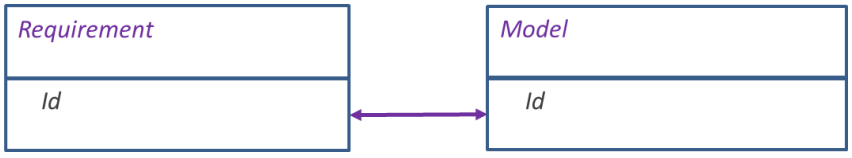
This MBSE context raises the question – that is still diversely apprehended today – wether the modeling approach should –at least partially- replace the conventional engineering method. So far the widely adopted solution is rather a “Cohabitation” solution [Bernard, 2011] where both a model and a set of requirements are managed in parallel, and consistency checking is managed through traceability links.

Both the specification and the validation of all the traceability links must be managed.

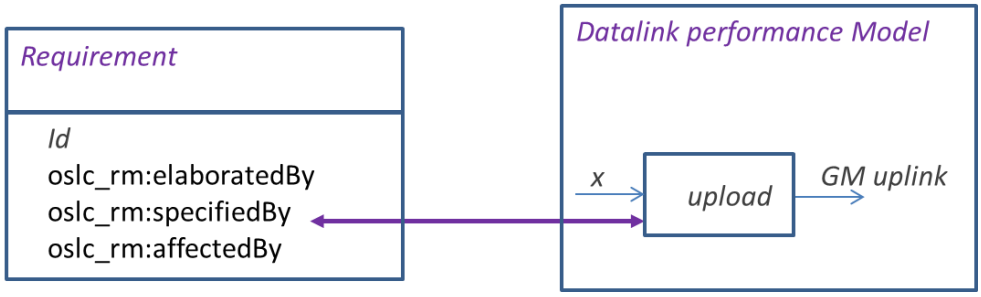
Traceability links should convey the meaning that the requirement is represented / modelled by the model. But the traceability “activity”, which is a manual human activity today, is hampered by the many problems related to the granularity of the information: is the requirement modeled by the model? Is it modeled by part of the model? Is the model refining the requirement? .... For what concerns the validation of a traceability link, as long as requirements are expressed in natural languages the validation of the traceability links remains a human activity.

Information granularity & treaceability

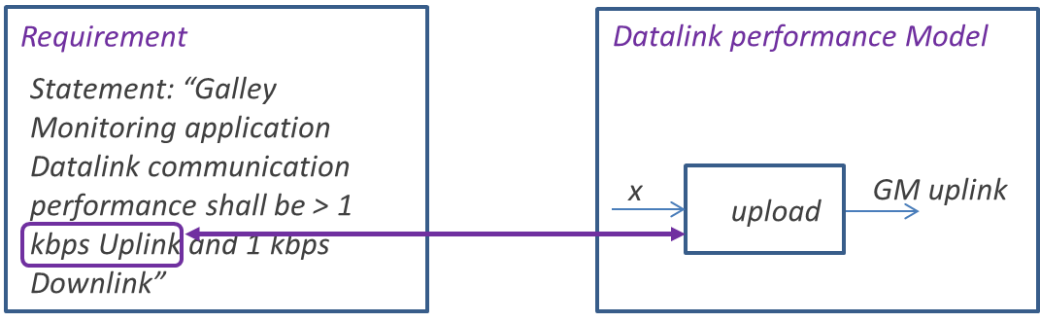
Different granularities of information exist that may be used to establish relations between requirements and models. A requirement is mandatorily referred to by a unique ID, and so should be a model. Simplest traceability between requirements and models can be based on ID level mapping principle (usually requirements are managed externally and refered by models).



Traceability links can be created between a requirement and an element of the model such as a diagram (e.g. Use case, IBD, activity diagram, sequence diagram, FFBD, etc.) or even diagrams’ sub-parts built from finer entities (block, ports, lines, links, items, and so on). OSLC Requirement resource http://open-services.net/ns/rm#Requirement entails a list of references that authorize this mechanism.



Finally inside a requirement's natural language statement, or inside a parametrized representation of a requirement (see below) one can isolate key values that are assessd by the models of the system under design. Enabling the traceability at this level requires that the system parameter name and physical dimension, but also the quantity expression and unit value are known and formalized or can be retrieved.

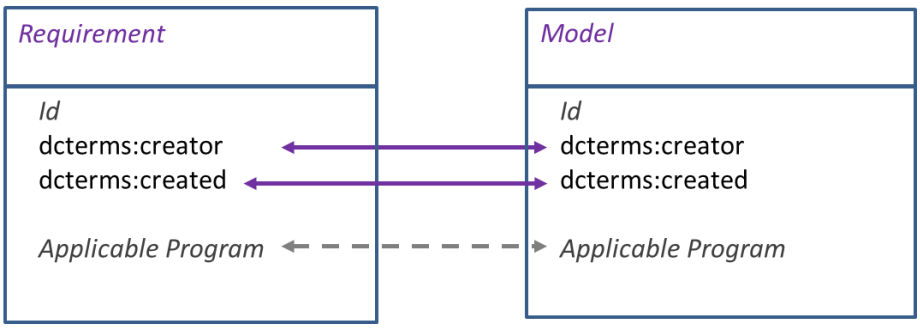


Other requirement <> model relationships

Tools which rely on proper database technology (e.g. IBM Rational DOORS Next Generation) instead of on file management; provide capability to manage requirements individually and also to define artifact types and their attributes.

A requirement object has a set of valued attributes for its statement, author, validation status, rational information, application context, etc. Any of such attributes provided it is uniquequely identified, could be referred to for creating other kinds of relationships (different from proper traceability) if relevant for given search or filtering purposes related to a given collaboration context. This requires that these kinds of attributes are treated as common metadata and used to characterize the various reference data and objects.

OSLC Requirement resource http://open-services.net/ns/rm#Requirement entails a list of requirement attributes or references that can similarily apply to the description of other resources and can be extended.



6.3 Semi-formal requirements specification and V&V

As already stated before (see §3.1), in order to convey the requirements the functions must be quantified. This quantification is simply referred to in a functional requirement. For example for the function "*Produce thrust*", the related performance requirement could be "150 kN". Thus functions are linked to performance attributes, referring to quantifiable physical properties (*temperatures, pressures, loads, forces, velocities, positions, moments*, etc.).

Objective of defining parametrized requirements is to better formalize and identify the quantifiable parameters and their associated quantified targets; for example the targeted performances associable to the system functions, and other quantifiable non-functional characteristics, *e.g. cost, size*. This objective is actually a fundamental enabler for linking requirements to models or for enabling quantitative impact analysis as required in some SP2 envisaged collaborative scenarios; even if, of course, many other (non parametrized) requirements will still co-exist.

Below are very simplified examples of parametrized requirements and alternative natural language representations.

#MTOW Range: [-;300]; #MTOW Unit: Ton	*The maximum takeoff weight of the A/C shall be less than 300 tons*
#Wing ice accretion thickness Range: [0;TBD]; #Wing ice accretion thickness Unit: mm	*The ice protection system shall prevent ice accretion on the wing greater than TBD mm*

They are simplified because their standalone interpretation would actually require the mention of the function and operational conditions for which the target has been set.

Parametric requirements are used in the Aerospace domain (Thalès, Rockwell-Collins, CNES) to support Product Line Engineering. Various tools implement parametric requirements management such as Cognition's Cockpit, Dassault Systems Requirements Central, PTC Integrity. To our knowledge, a technology like COGNITION Cockpit (http://cognition.us/cognition-cockpit) is representative of the state of the art as a collaborative platform for managing parametrized requirements.

They actually relies on human manual edition of the requirement critical parameters and target values (range + unit) in a specification template.

Even if it may sound an enormous job to reference and maintain all parameters, it is actually done for many of them. The complete set or numerical messages exchanged between all aircraft systems are referenced and maintained thanks to a System Interfaces Definition (SID) process and database. Multidisciplinary trade-off capability requires the sharing and exchange of the key parameters that need to be shared by disciplines and this is currently addressed in Airbus through the internal eCascade project above mentioned in 3.1 as well as through former CRESCENDO and SDM and current TOICA projects.

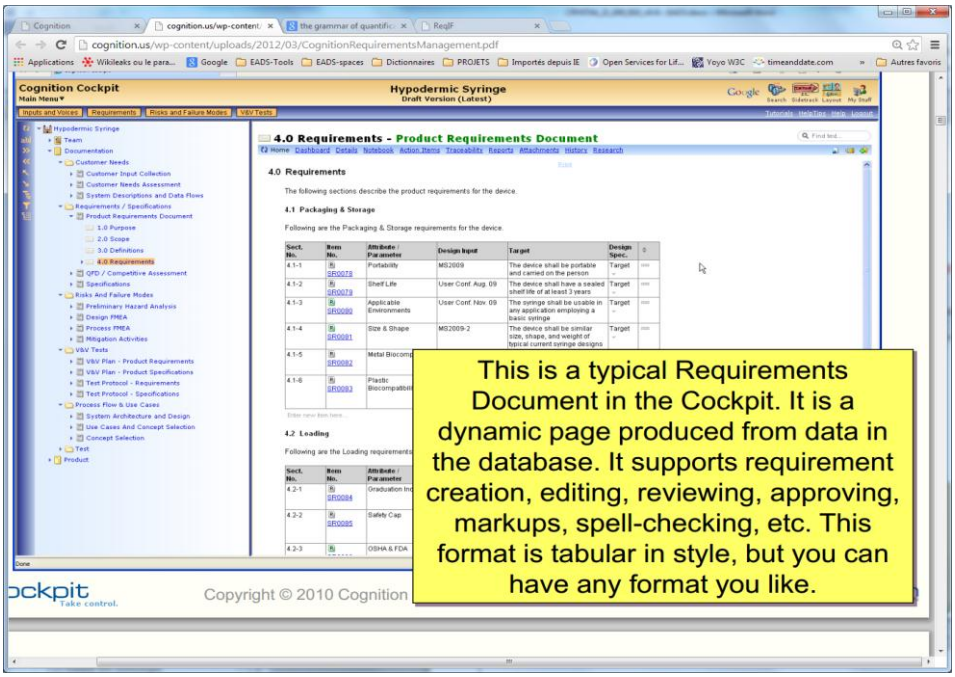


Figure 6-2: IHM screenshot of COGNITION cockpit

For what concerns function reference, backgroung already exist. Reference lists were established, for example Safety Analyses discipline has long described lists of functions and structure the management of their data accordingly. Reference functional breakdown exist for A/C and system functional analysis (ALFA). The A350 requirement database and document cascade structures are based on a functional breakdown. Other initiatives have also been taken to homogenize the way to either express functions in natural language ([Kipper, 2006.] [Hirtz, 2002]), or model them in formalized specifications ([Osaka] [SVBR]),

Alternative approach toward the same objective consists in identifying key parameters and target value ranges within the requirements natural langage statements. This could allow dynamically retrieving the desired information in natural language statements rather than requiring static parameter entities management (data-base view). This requires complex parsing and pattern recognition to retrieve and interpret the parameters names and quantity expressions whitin the natural language statements. Technologies such as TXM illustrated below (an XML-TEI encoded corpora compatible analysis platform for text mining allowing CQL query language [Heiden 2010]), can be used for this purpose. SP6 WP6.7 technical partners bring in the CRYSTAL project similar capabilities.

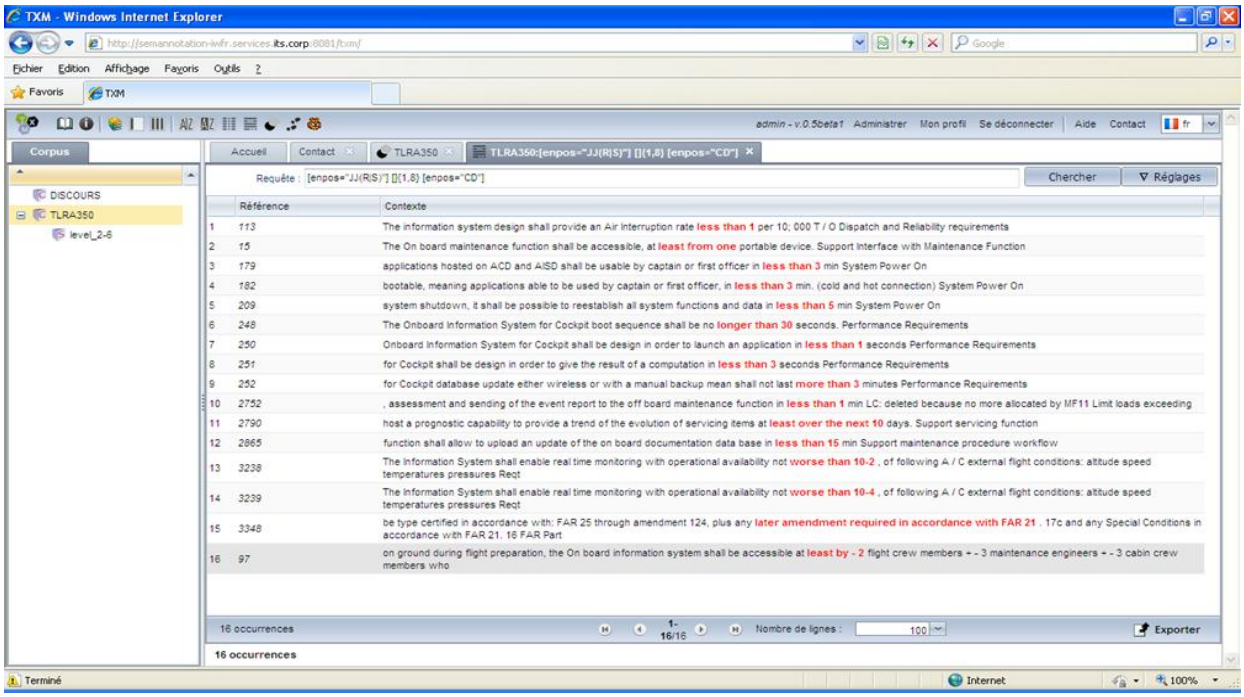


Figure 6-3: TXM, an XML-TEI encoded corpora compatible analysis platform for text mining

The two above describe approaches would benefit from a shared definition for physical dimensions, quantities, units, functions, etc. that could be considered in the scope of the Domain ontology.

Finally, the Figure 6-4 below foresees additional knowledge regarding parameters, here called “transfer function” to make explicit the relationships between the parameters.

Methods to describe such kind of knowledge (algebraic constraints, integrity rules) were studied in software area in the field of Domain analysis and Domain Feature modeling ([Kyo, 1990], [Arango, 1989]). Examples are interestingly applied in [Moros, 2008] or [Siegmund,] to the formalizing of variability in requirements in order to support their reuse for product line management. The example above means that the value of one variable is equal to the sum of the value of two other variables at any time. Such mathematical knowledge must include the facts that a variable stands for a certain value that may or may not change in time and that possibly has a certain dimension. Without this knowledge the formula is hardly interpretable.

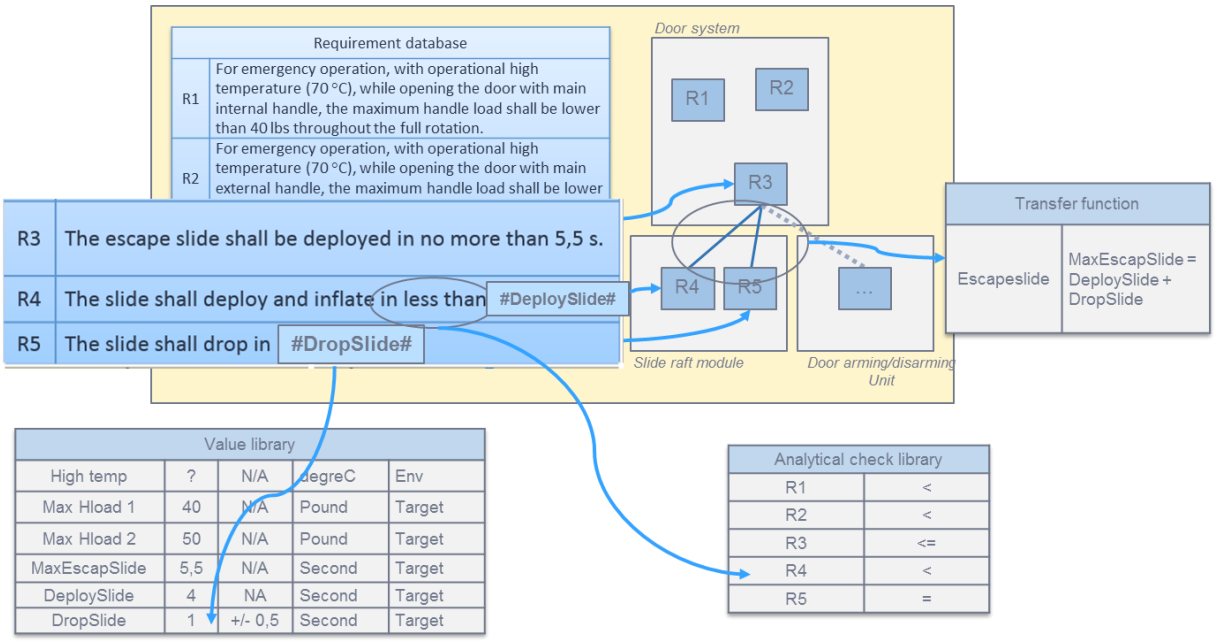


Figure 6-4: Retrieving parametrized information within a requirement

Therefore a shared and richer information model is needed in order to enable automatic retrieval, exchange, or integrity checks over these data. This represents another possible aspect of formal ontological descriptions in Engineering ([Borst,1997],[], [Gruber, 1994b]]).

It will be a work in the coming T209.02 specification task to analyse which elements of the identified inputs and resources can be interestingly integrated in the domain ontology, depending on the foreseen services.

7 Support tools and Technologies

This chapter presents some tools and platforms that are considered suitable for supporting the integration of domain ontology concepts in the envisaged CRYSTAL reference technology platform.

7.1 Semantic WEB Technologies

In this section we introduce the basic concepts relative to the semantic web; after a general introduction, to explain the goal of the semantic web, we examine one by one the building blocks, considering both the technologies and their applications.

7.1.1 W3C Standards

The World Wide Web Consortium (W3C) is created in 1994. It is the main international standards organization for the World Wide Web (abbreviated WWW or W3). The W3C proposes and promotes a quite long list of standards related to web architecture, XML technologies, web services, browsers and authoring tools, etc. The description of these is out of the scope of this document which focuses on semantic technologies, but can be found at http://www.w3.org/standards/.

Its object is the development of technologies (specifications, recommendations, software and tools) and the proposing of standards for the growth and use of the Web.

The web traditionally permits to access data in a form suitable for human consumption. Regular users can visit websites composed by HTML pages, Flash or Applet applications. They can play videos, listen to audio snippets embedded in the pages, or observe images contained there. If they want to obtain more information on a related concept they can use a link present in the page.

The information is given to the user in an already presentable form, ready to be used. On the other end the data can be just consumed under the particular representation that the data owner chose to expose. It cannot be directly processed for goals which were not envisioned by the data provider.

The semantic web instead provides access to data in machine processable formats. In this way applications can expose, access and elaborate data. Mesh-up and combinations of services become possible. In the words of Tim Berners-Lee:

> *The first step is putting* data *on the Web in a form that machines can naturally understand, or converting it to that form. This creates what I call a Semantic Web – a web of data that can be processed directly or indirectly by machines.* [Weaving the web, p. 191]

A relevant example of the emerging semantic web is given by DBPedia:

> *DBpedia is a crowd-sourced community effort to extract structured information from Wikipedia and make this information available on the Web. DBpedia allows you to ask sophisticated queries against Wikipedia, and to link the different data sets on the Web to Wikipedia data. We hope that this work will make it easier for the huge amount of information in Wikipedia to be used in some new interesting ways. Furthermore, it might inspire new mechanisms for navigating, linking, and improving the encyclopedia itself.*[1]

In other words DBpedia is the transposition in the semantic web of what Wikipedia is in the traditional web: the same contents which Wikipedia provides for human beings, DBpedia exposes in format which is suitable to be processed by machines. DBpedia plays also a pivotal role in the semantic web because different services are able to interoperate with it, combining third-part information with the content of DBpedia.

This concept is depicted in the Figure below, extracted from http://lod-cloud.net/versions/2011-09-19/lod-cloud.html .

[1] From http://dbpedia.org/about

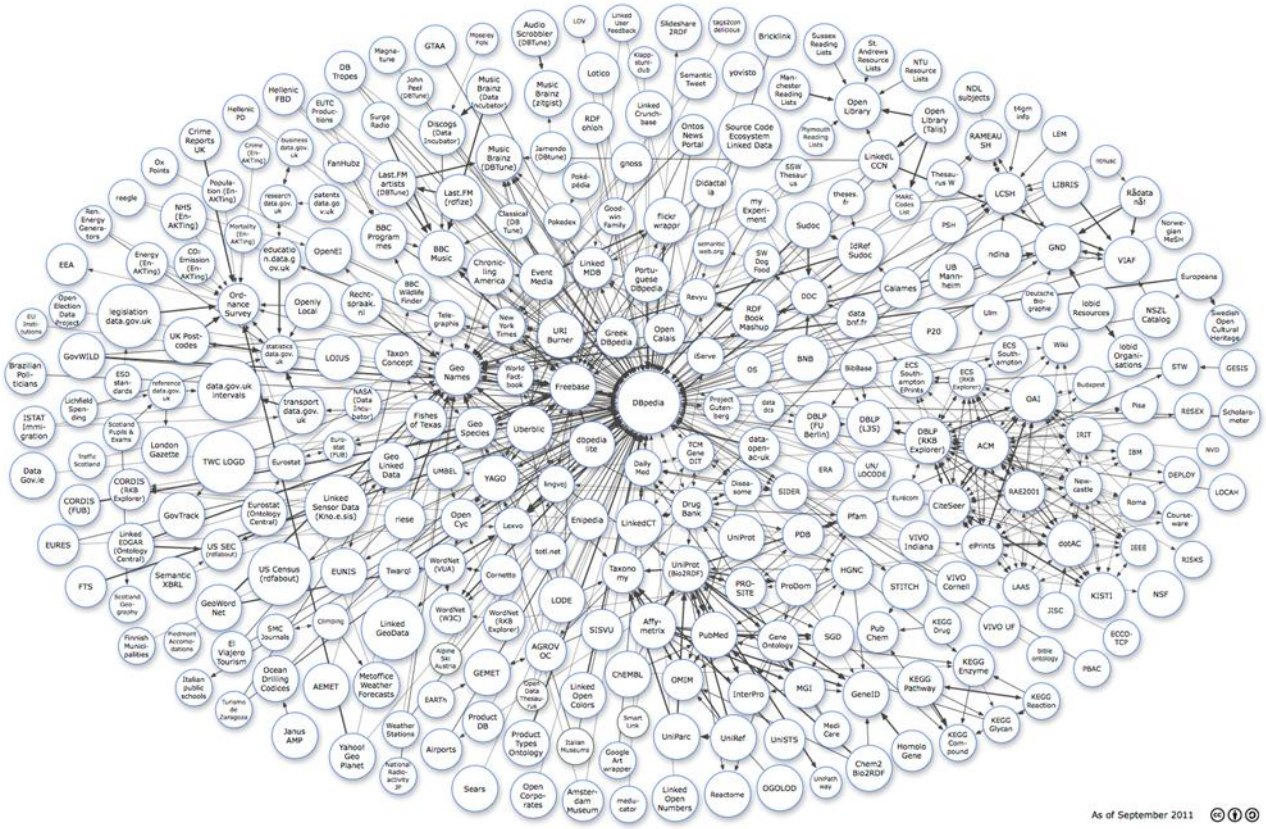


Figure 7-1: DBpedia concept

This architecture follows the principle of Linked Data as presented by Tim Berners-Lee: data is not only exposed in machine readable formats but it is inter-connected and navigable. As the traditional web is based on links which permit to navigate across related concepts, the semantic web employs mechanism which permits to associate different sets of data, depicting different concerns, to the same entity and describe connections between entities. In this way, for example, a website from the Italian government could present official data about the city of Torino: the number of inhabitants, the region which it is part of, the current Mayor, etc. Another website could define the list of famous artists related to the city because they were either born or dead there. Both the sites would refer to the same entity (the city of Torino), describing different aspects. Moreover from this definition it would be possible to navigate to connected entities (e.g., the containing region or the related artists), retrieve data associated with those entities.

A big advantage is the possibility to realize applications unforeseen from the original actors who provided the data. Consider the scenario in which a website A is offering the data about the different exhibitions offered in all the cities of one country, while another website B offers data

about the mayor governing a certain city. By combining data offered by A and B we could evaluate which mayors were able to better operate in the cultural area by promoting the larger number of successful exhibitions in respect to the population of the city governed. An application automatically mixing the data obtained from these different sources could be able to provide this new information. Moreover, as the original sources were updated also the derived information could be automatically updated.

While offering data to human the provider can take advantage of the ability of the user to interpret that information using context and common sense. When the target is a machine, the data has to be presented with different characteristics that we are going to examine in the rest of the chapter.

7.1.2 Formalisation of knowledge

A summary of what is needed to make data suitable for processing and inter-related includes:

- Unambiguous names for resources (that may also bind data to real world objects): URIs
- A common data model to access, connect and describe the resources: RDF
- Query the data: SPARQL
- Define common vocabularies: RDFS, OWL
- Reasoning logics: OWL

A specific W3C standard family concerns the semantic web which main constituents are those mentioned above. Linked data in RDF, vocabularies or thesaurus typical in SKOS or OWL, query languages such as SPARQL, and inference rules and reasoners (ref. http://www.w3.org/standards/semanticweb/). The integration of these languages is often represented by the Web Technology Stack picture, a version of which is reproduced below.

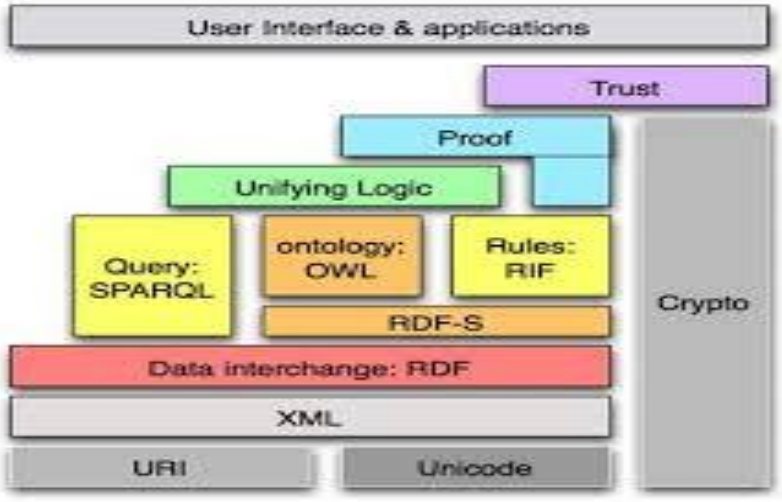


Figure 7-2: Web Technology Stack

In the rest of the chapter we will examine these building blocks separately.

7.1.2.1 URIs: identify entities unambiguously

When using a natural language (e.g., French or Spanish) often the meaning of a term depends on the context where it is used. Human beings are able to use context and previously acquired information to discriminate among the possible meaning of a term and pick up the one intended by the speaker.

To permit machine processing of information we need instead to specify unambiguously to which entity we are referring. For example the term “Crystal” could refer either to an European project or to a solid.

Consider another example: the term “Paris”. While the most well-known entity associated to this term is the city, capital of France a lot of other entities could be associated to that name. Among them two cities in Canada, around twenty villages or communities, in the USA, a city in Denmark, a settlement in the Kiribati island, a Roman actor (http://en.wikipedia.org/wiki/Paris_%28actor_under_Domitian%29), a saint http://en.wikipedia.org/wiki/Saint_Paris and other people.

So if you ask someone in a bar of the Linn County, Iowa, where Paris is, probably they will explain which road to take to reach it in a matters of minutes, sending you to the community founded in 1841 in that state. If you want to talk about the “French Paris” in that same bar, you probably have to specify it explicitly, while in France or in Europe in general, the French city is the intended entity

associated with that term. Unless you are talking about roman actors or saints, in which case the entities we referred to before could be considered.

To avoid this confusion unambiguous names are used in the Semantic Web. In particular Uniform Resource Identifiers (URI) are used. URI are composed by a scheme name and a scheme-specific part. Well known examples of URI are http URLs or ISBN codes. URI are classified either as URL or URN, but in the context of this work we do not consider important to differentiate between them. The important concept is that one URI is associated exactly to one entity, while the contrary is not true, it means different URI could identify the same entity.

In the following table we report some examples of URI.

Type	Scheme	Scheme-specific	Complete
URL	http	en.wikipedia.org	http://en.wikipedia.org
URN	isbn	0-486-27557-4	isbn:0-486-27557-4
URL	file	somedir/myfile.pdf	file://somedir/myfile.pdf
URN	tag	govtrack.us,2005:congress/senators/frist	tag:govtrack.us,2005:congress/senators/frist

7.1.2.2 RDF – Resource description Framework

The Resource Description Framework (RDF) is a W3C specification, created in 1999, to describe metadata. It initiates the structuring of Web information. The objective is to enable automatic processing of the information and to promote interoperability. RDF is based on XML. It uses URIs as a naming scheme in order to disambiguate terms with different meanings, as explained in the previous section.

Entities can be described in RDF using triplets. Triplets are constituted by a subject, a predicate and an object.

Examples of facts that can be expressed in RDF are presented in the next table.

Subject	Predicate	Object
Crystal	Is-a	European project
Crystal	Is-a	solid

Tom	Is-tall-meters	1.8
Tom	Is-years-old	20

Looking at the table we can see a problem we have already discussed in the previous section: by using arbitrary terms we have to face possible ambiguity. Therefore RDF adopts URI for the subject and the predicate. For the object both URI and literals can be employed.

For example we could restate the the example in the first line of the table as:

http://www.crystal-artemis.eu/	rdf:type	<URI-for-european-project>

We could use the URL associated to the project as the subject, a URI identifying the property “is a” (rdf-type) for the property while we would need an URI identifying the entity “European Project” to be used as object. We are not aware of a well-known URI defining this particular concept so we could possibly define it, for example adopting an unused scheme and defining our own URN or find a suitable URL.

In the third and fourth examples shown in the table we employed two literals (an integer and a float number). Other examples of literals are Strings or Dates.

Other examples extracted from DBPedia:

```
<http://dbpedia.org/resource/Turin>
     <http://xmlns.com/foaf/0.1/name>
     "Turin"@en .
<http://dbpedia.org/resource/Turin>
     <http://www.w3.org/2002/07/owl#sameAs>
     <http://es.dbpedia.org/resource/Tur\u00EDn> .
<http://dbpedia.org/resource/Turin>
     <http://dbpedia.org/ontology/wikiPageExternalLink>
     <http://www.buddies.it> .
<http://dbpedia.org/resource/Turin>
     <http://dbpedia.org/property/region>
     <http://dbpedia.org/resource/Piedmont> .
<http://dbpedia.org/resource/Turin>
     http://dbpedia.org/property/mayorParty
     <http://dbpedia.org/resource/
               Democratic_Party_(Italy)> .
```

RDF can be serialized in different ways: XML, N3 notation, JSON. The serialization format can be chosen freely depending on the availability of libraries for the platform considered and performance constraints.

RDF is thought to be used in a decentralized way to fit the nature of the web, in which every participant can contribute without supervision. As consequence of the decentralized nature of RDF we cannot do assumptions about the facts expressed: a contributor A could express a fact which is not coherent with what was expressed by a contributor B.

In practice, the decentralized nature of RDF has these consequences:

- Allow anyone to make statements about any resource,
- No guaranteed completeness,
- No guaranteed consistency.

RDF permits also to assign an URI to single RDF statements to describe them. For example we could specify the source providing a certain statement or a level of confidence. In some way it permits to associate metadata to statements.

Consider this statement:

```
exproducts:item10245 exterms:weight "2.4"^^xsd:decimal .
```

It is identified by the URI exproducts:item10245. We can now express statements referring to that statement:

```
exproducts:triple12345 rdf:type rdf:Statement .
exproducts:triple12345 rdf:subject exproducts:item10245 .
exproducts:triple12345 rdf:predicate exterms:weight .
exproducts:triple12345 rdf:object "2.4"^^xsd:decimal .
exproducts:triple12345 dc:creator exstaff:85740 .
```

7.1.2.3 Query the data: SPARQL

In the previous section we have seen how it is possible to define data, in this section we take a look at how it is possible to use this data.

The SPARQL language is a language to express queries on a base of knowledge composed by RDF statements. It stands for SPARQL Protocol and RDF Query Language. This language was endorsed by Tim-Barners Lee himself and its regarded as a fundamental building block of the Semantic Web.

It is way similar to the well-known SQL language, while it was created to fit the specificity of the RDF knowledge bases on which it is used. Queries consist of a select clause, which specify the data we want the query to retrieve. The body of the query is a set of RDF triplets in conjunction or disjunction, optionally followed by a filter clause.

Let's consider a first example:

```
SELECT ?name, ?surname WHERE {
  ?person a <http://dbpedia.org/ontology/Person> .
  ?person   <http://it.dbpedia.org/property/nome>    ?name .
  ?person   <http://it.dbpedia.org/property/cognome> ?surname .
  ?person   <http://it.dbpedia.org/property/nazionalità> "italiana"@it .
  FILTER( isLiteral(?name) AND isLiteral(?surname) )
}
```

This query permits to retrieve the name and surname of all Italian persons present on DBPedia.

This query returns a list of pairs, each pair being composed by a name and a surname *(?name, ?surname)*. These values are present in the triplets contained in the body of the query. The body of the query is composed by four RDF triplets that have to be satisfied all at the same time. The first one:

```
?person a <http://dbpedia.org/ontology/Person> .
```

This statement collects all the entities of type Person in the list *named ?person*. The predicate “a” is a shortcut for *rdf:type*, which means the entity is part of a given category (more on this topic will be explained in the section about RDF Schema).

```
?person   <http://it.dbpedia.org/property/nome>    ?name .
?person   <http://it.dbpedia.org/property/cognome> ?surname .
```

For each element that we collected in the list ?person we select the name and the surname. In practice the SPARQL engine looks for statements where the subject is contained in the ?person list, the predicate is the one specified (http://it.dbpedia.org/property/nome or http://it.dbpedia.org/property/cognome) and collect the objects of these statements.

```
?person   <http://it.dbpedia.org/property/nazionalità> "italiana"@it .
```

Finally this last statement constraint the persons selected to be of Italian nationality only.

The filter section in this case specifies that we are interested in collecting literals. It could be possible that the knowledge base contains the definition of a surname as an entity and therefore it could contain statements having as predicate http://it.dbpedia.org/property/cognome and as object an entity instead of a simple literal.

```
SELECT ?name, ?surname WHERE {
  ?person a <http://dbpedia.org/ontology/Person> .
  ?person   <http://it.dbpedia.org/property/nome>     ?name .
  ?person   <http://it.dbpedia.org/property/cognome>  ?surname .
  ?person   <http://it.dbpedia.org/property/nazionalità> "italiana"@it .
  ?person   <http://it.dbpedia.org/property/attività> "artista"@it .
  ?person   <http://it.dbpedia.org/property/sesso>    "F"@it .
  ?person   <http://dbpedia.org/ontology/birthYear>   ?birth .
  FILTER( isLiteral(?name) AND isLiteral(?surname) AND ?birth >= "1972-01-01"^^xsd:date)
}
```

This is an example of query for retrieving the name and surname of all Italian female artists born after 1972 and present on DBPedia.

In this other query we filter on data specifying it should be successive to a reference value. SPARQL permits to express basic constraints on a few primitive types, including string, numbers and dates. It is even possible to specify regular expressions to filter strings.

There are publicly available SPARQL endpoints which permit to insert SPARQL queries through a web interface and visualize the results. One of them is: http://dbpedia.org/sparql.

7.1.2.4 Define common vocabularies: RDFS, OWL

Until now we have seen how it is possible to specify data and query this data. Now we will see how it is possible to define common structures for this data and common vocabularies. The Linked Data works when the same terms are used to refer to the same entities with a common structure. If we considered previous examples about query DBPedia for persons with a given first and family name they worked because all the persons present in DBPedia are described in this way. The same queries would not work if instead a property describing the full name of a person (i.e., the concatenation of the first and family name) would have been used across DBPedia. We need to agree on common structures and use them consistently to later be able to use these structures to

query the dataset. In this section we examine how we can define those structures, which we call ontologies.

To describe ontologies there are different technologies available which permit to specify different levels of detail. We briefly introduce RDF-Schema and OWL.

RDF Schema

RDF Schema is a set of RDF classes which are uses to organize RDF entities according to basic principles. RDF Schema permits to specify that a given entity is part of a particular class. This is done using a simple RDF triplet in which the subject is the entity, the predicate is the *rdf:type* and the object is the class.

A class with a special meaning is *rdfs:Class* it is used to specify which entity can be used as a class. *rdfs:Resource* instead includes every possible entity. From this information we can derive the both instances and classes are entities and can be mostly treated homogeneously. This simplifies the language[2].

Other relevant classes which are directly specified by the standard are:

rdfs:Literal – this class includes all literal values (e.g., strings and integer).

rdfs:Datatype – the class of datatypes. *rdfs:Datatype* is both an instance of and a subclass of *rdfs:Class*. Each instance of *rdfs:Datatype* is a subclass of *rdfs:Literal*.

rdf:XMLLiteral – the class of XML literal values. *rdf:XMLLiteral* is an instance of *rdfs:Datatype* (and thus a subclass of *rdfs:Literal*).

rdf:Property – the class of properties (described in the following text).

Properties are particular entities which can be used as predicate in RDF triplets to specify qualities or relations of other entities. For example to specify the class of which an entity is part, the property *rdf:type* can be used.

The property *rdfs:subClassOf* allows to declare hierarchies of classes. It is possible to specify that a class is a subclass of another class. The relation is transitive. In the same ways does exist the property *rdfs:subPropertyOf*, which permits to specify hierarchies between properties.

[2] It is in general considered a good property, because it permits to a certain extent to express the meta-meta-model (the basic elements of RDF-Schema) in the terms of itself, without relying on entities not referable in the language itself.

For more details prefer refer to the defining document, available at: http://www.w3.org/TR/rdf-schema/.

7.1.2.5 OWL: Web Ontology Language

OWL becomes a W3C recommendation in 2004. It is not a single language but rather a family of related languages to formalize ontologies. All these languages are serializable as RDF.

There are different languages which permit to express ontologies with different levels of complexity. In this brief introduction we will not focus on the differences among the different levels but we will instead provide a general feeling about the possibilities offered by the language family as a whole.

In general OWL is intended to be compatible and built on existing RDF Schemas. Hence, it can be used to enhance existing “basic” ontologies defined through RDFS by specifying additional constraints. In this sense OWL can be considered a semantic extension of RDFS.

By principle, OWL is based on the same assumption on which is based the Semantic Web and RDF in particular: only what you explicitly know is defined, while anything else is not necessarily false but it is undefined. Moreover it is possible the existence of contradictory information. OWL ontologies are intended to be used in a distributed environment (the web), therefore they are designed to be combined and related, therefore each OWL ontology hase to be explicitly identified by an URI.

In OWL every possible class implicitly extends *owl:Thing*, while the class owl:Nothing does not contain any element. In OWL ontologies is possible either to define classes and single instances as well.

Constraints can be applied to classes using the concept of *owl:Restriction*. A possible restriction can enforce a minimal or maximal cardinality for a property. For example we could specify that a football team should have at least eleven players.

It is possible to specify properties, with their domain (the possible subjects) and range (the possible objects). Also properties, as classes, can be organized in hierarchies.

In OWL, a class defines a set of individuals that have similar properties. A specialization relationship allows creating a hierarchy of classes: a sub-class inherits the properties of its parent class, and it refines the description with additional properties. A sub-class defines a more specialized set of individuals.

Properties can have specific characteristics, which have effects on reasoning (see the next section).

Transitive: formally it means that for a property P:
P(x,y) and P(y,z) implies P(x,z)

Symmetric:
P(x,y) iff P(y,x)

Functional:
P(x,y) and P(x,z) implies y = z

InverseOf in respect to a property Pb

P(x,y) iff Pb(y,x)

InverseFunctionalProperty:

P(y,x) and P(z,x) implies y = z

Ontologies can be combined. The web itself is devised as a collaborative platform and the OWL standard strictly adheres to this principle. To favor collaborative combinations of ontologies it is possible to combine different ontologies specifying mappings between classes and properties of different ontologies. For example the class team of an ontology A could be mapped on the class footballTeam of an ontology B. In that case whatever was stated on the class team would be also valid for the class footballTeam and viceversa.

It also possible to declare an relation of equivalence between instances. It is equivalent to define aliases for same term. We could define the instance *myFavoriteFootballTeam* to be mapped on *TorinoFC*, for example.

It is possible to define a class as an intersection between classes or other collection of instances that respect certain constraints. Consider for example this construct[3]:

```
<owl:Class rdf:ID="WhiteWine">
  <owl:intersectionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#Wine" />
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasColor" />
      <owl:hasValue rdf:resource="#White" />
    </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>
```

Other possible set operations on groups of classes are the union:

[3]This and other examples could derive from W3C standards and related documentation

```
<owl:Class rdf:ID="Fruit">
  <owl:unionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#SweetFruit" />
    <owl:Class rdf:about="#NonSweetFruit" />
  </owl:unionOf>
</owl:Class>
```

And the complement operation:

```
<owl:Class rdf:ID="NonConsumableThing">
   <owl:complementOf rdf:resource="#ConsumableThing" />
  </owl:Class>
```

Similar to complement (but more relaxed) it is the constraint *disjointWith*:

```
<owl:Class rdf:ID="Pasta">

  <rdfs:subClassOf rdf:resource="#EdibleThing"/>

  <owl:disjointWith rdf:resource="#Meat"/>
  <owl:disjointWith rdf:resource="#Fowl"/>
  <owl:disjointWith rdf:resource="#Seafood"/>
  <owl:disjointWith rdf:resource="#Dessert"/>
  <owl:disjointWith rdf:resource="#Fruit"/>
</owl:Class>
```

Another common use case in defining ontologies is the necessity to specify classes through explicit enumeration of all possible components:

```
<owl:Class rdf:ID="WineColor">
  <rdfs:subClassOf rdf:resource="#WineDescriptor"/>
  <owl:oneOf rdf:parseType="Collection">
    <owl:Thing rdf:about="#White"/>
    <owl:Thing rdf:about="#Rose"/>
    <owl:Thing rdf:about="#Red"/>
  </owl:oneOf>
</owl:Class>
```

7.1.2.6 Reasoning logic: OWL

In the previous section we have seen as we can define data and their structures, listing a set of restrictions and assumptions about the data. Using these assumptions, we can derive conclusions which permit to produce knowledge which was not explicitly present in the original knowledge base. The set of tools which perform these derivations are named semantic reasoners. These kind

of tools permit to infer logical consequences from a set of asserted facts, which are considered axiomatic truths. Commonly they are based on first-order predicate logic.

The most well-known semantic reasoners are based on OWL entailment rules. Among them we like to cite Pellet4 and Protégé5.

Reasoning logic permits to infer new knowledge from existing facts. For example if we define a bus driver as a person who drives bus, and we derive a driver as a person who drives a vehicle, from knowing that a bus is a vehicle we can derive that a bus driver is a driver. These facts are expressed in OWL in this way:

```
Class(a:bus_driver complete intersectionOf(a:person
  restriction(a:drives someValuesFrom (a:bus))))

Class(a:driver complete intersectionOf(a:person
  restriction(a:drives someValuesFrom (a:vehicle))))

Class(a:bus partial a:vehicle)
```

From these facts it derives that every affirmation which applies to drivers applies also to bus drivers.

In the context of this introduction to the state of the art we do not consider useful to explain the detailed rules semantic reasoners employ to derive new facts. They are however based on the semantic associated with the properties defined in the standards, in particular OWL.

7.2 Collaborative Platforms

This section addresses tools and platforms that may help in creating, editing and viewing ontologies. It shall be possible to support the building, publishing and usage of created ontologies.

7.2.1 W3C technologies

No specific development in CRYSTAL SP6 is foreseen for ontology editing (apart from the Knowledge manager from Reuse company which is dedicated to the specific ‘linguistic’ ontology).

Also no dedicated mean is brought in by the WP209 partners for edition or reasoning. In this context we are considering tools and technologies currently available from the W3C community and the open source community.

[4]https://github.com/clarkparsia/pellet
[5]http://protege.stanford.edu/

In particular, we are considering the “**Protégé**” platform

Protégé is a free, open-source platform that provides user community with a suite of tools to construct domain models and knowledge-based applications with ontologies. At its core, Protégé implements a rich set of knowledge-modeling structures and actions that support the creation, visualization, and manipulation of ontologies in various representation formats. Protégé can be customized to provide domain-friendly support for creating knowledge models and entering data. Further, Protégé can be extended by way of a plug-in architecture and a Java-based Application Programming Interface (API) for building knowledge-based tools and applications.

The Protégé platform supports two main ways of modeling ontologies:

The Protégé **Frames editor** enables users to build and populate ontologies that are frame-based, in accordance with the Open Knowledge Base Connectivity protocol (OKBC). In this model, an ontology consists of a set of classes organized in a sub-sumption hierarchy to represent a domain's salient concepts, a set of slots associated to classes to describe their properties and relationships, and a set of instances of those classes - individual exemplars of the concepts that hold specific values for their properties.

The Protégé **OWL editor** enables users to build ontologies for the Semantic Web, in particular in the W3C's Web Ontology Language (OWL). Given such an ontology, the OWL formal semantics specifies how to derive its logical consequences, i.e. facts not literally present in the ontology, but entailed by the semantics. These entailments may be based on a single document or multiple distributed documents that have been combined using defined OWL mechanisms (see the OWL Web Ontology Language Guide).

The editor in synthesis allows to:

- Load and save OWL and RDF ontologies.
- Edit and visualize classes, properties, and SWRL rules.
- Define logical class characteristics as OWL expressions.
- Execute reasoners such as description logic classifiers.
- Edit OWL individuals for Semantic Web markup.

Here below examples for the editor’s user interface are shown.

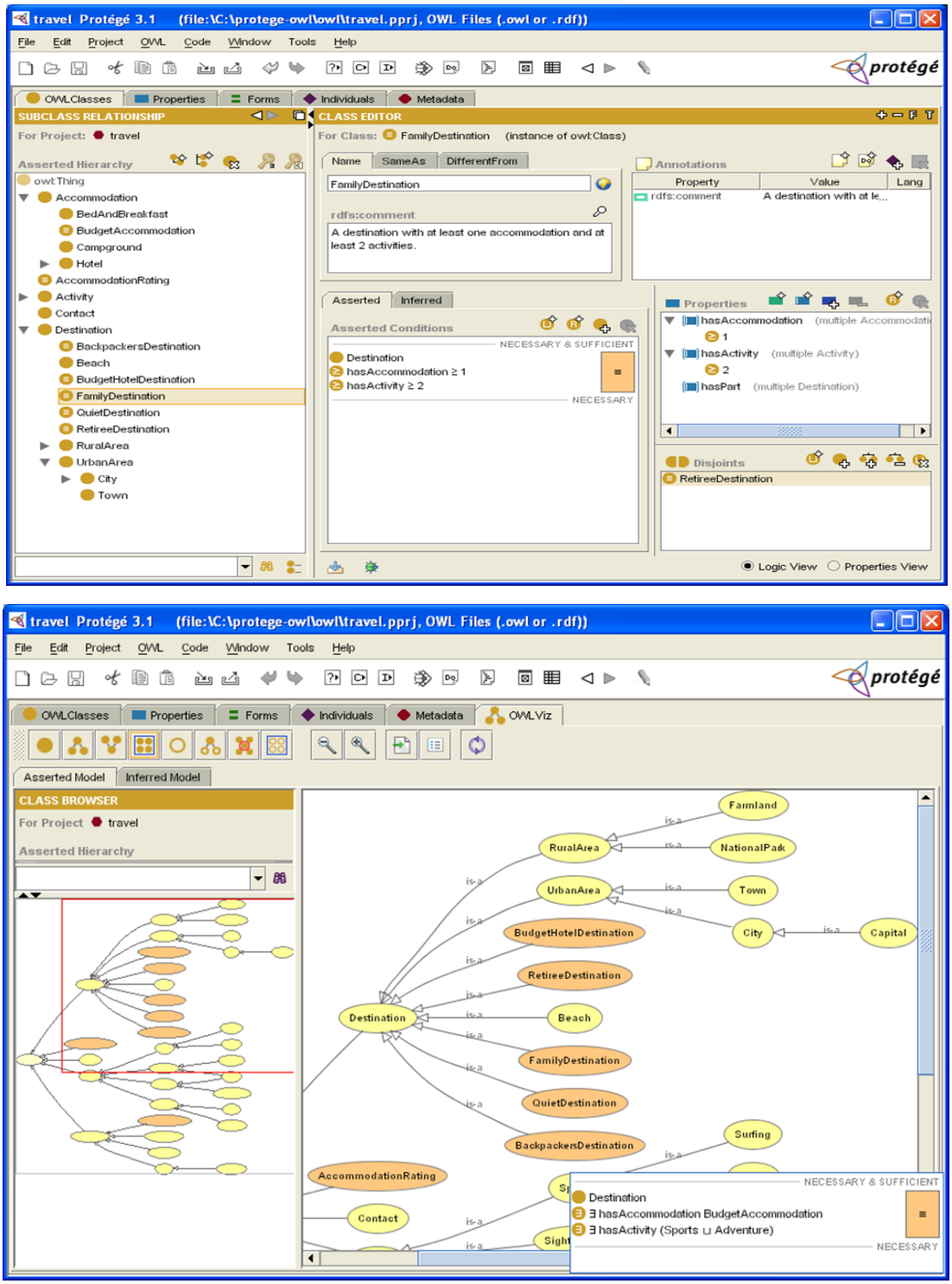


Figure 7-3: GUIs examples from Protégé editor

Further information about Protégé-OWL is available in the documentation section of the Protégé-OWL Web site, including the popular Protégé OWL Tutorial and the Protégé-OWL FAQ.

7.2.2 Current / envisioned usage of ontology in RTP platform

In order to build a platform support for integrating ontologies we can think about infrastructures such as the OpenLink Virtuoso.

A first concept for its adoption is shown in the picture below.

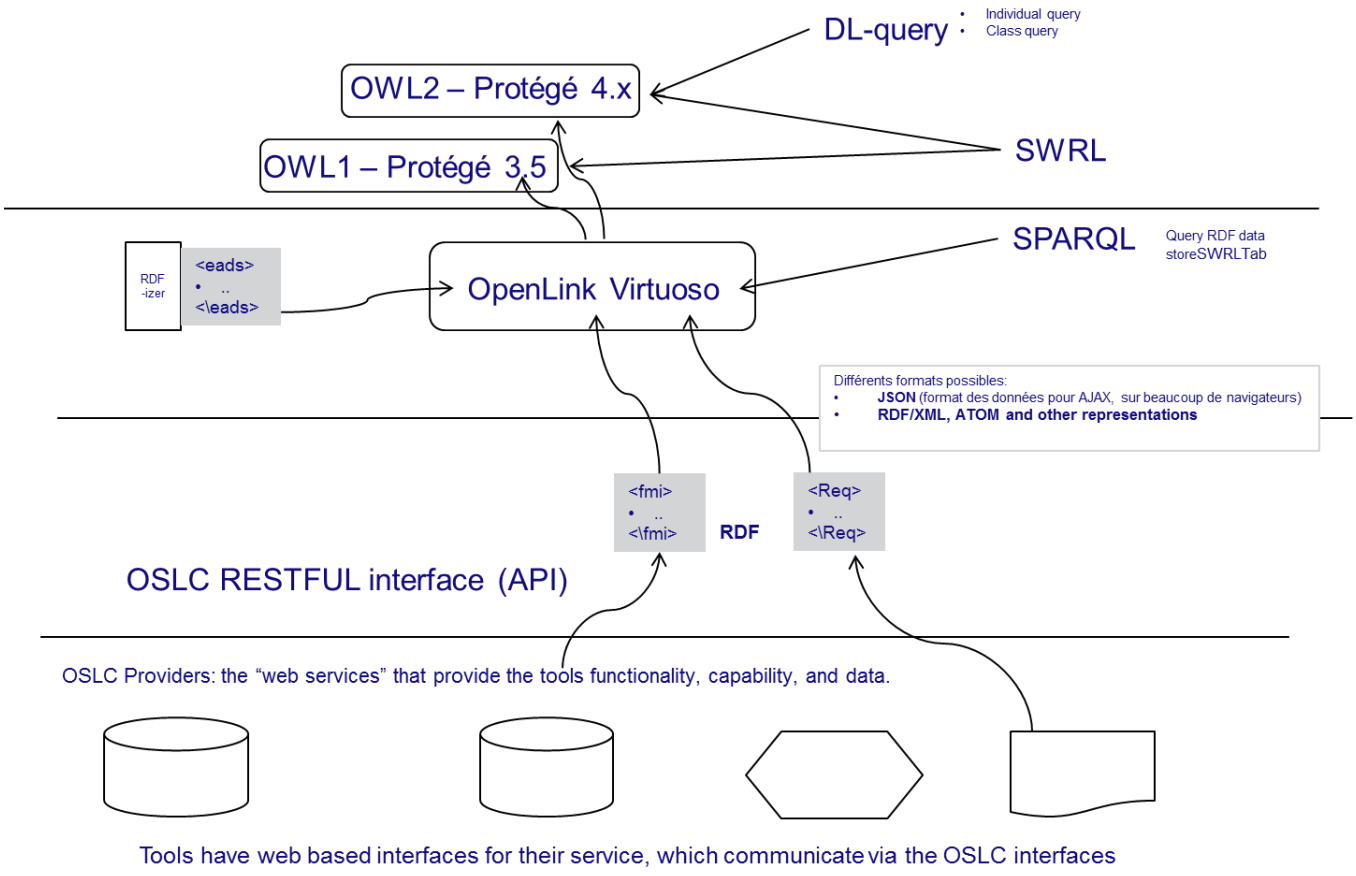


Figure 7-4: using OpenLink Virtuoso

An overview about Virtuoso and its capabilities is provided here below.

Virtuoso Universal Server is a middleware and database engine hybrid that combines the functionality of a traditional RDBMS, ORDBMS, virtual database, RDF, XML, free-text, web application server and file server functionality in a single system. Rather than have dedicated servers for each of the aforementioned functionality realms, Virtuoso is a "universal server"; it

enables a single multithreaded server process that implements multiple protocols. The open source edition of Virtuoso Universal Server is also known as **OpenLink Virtuoso**. The software has been developed by OpenLink Software.

The hybrid server architecture of Virtuoso enables it to offer server functionality that covers the following areas:

- Relational Data Management
- RDF Data Management
- XML Data Management
- Free Text Content Management & Full Text Indexing
- Document Web Server
- Linked Data Server
- Web Application Server
- Web Services Deployment (SOAP or REST)

Here below a schema for its general architecture and relations is presented

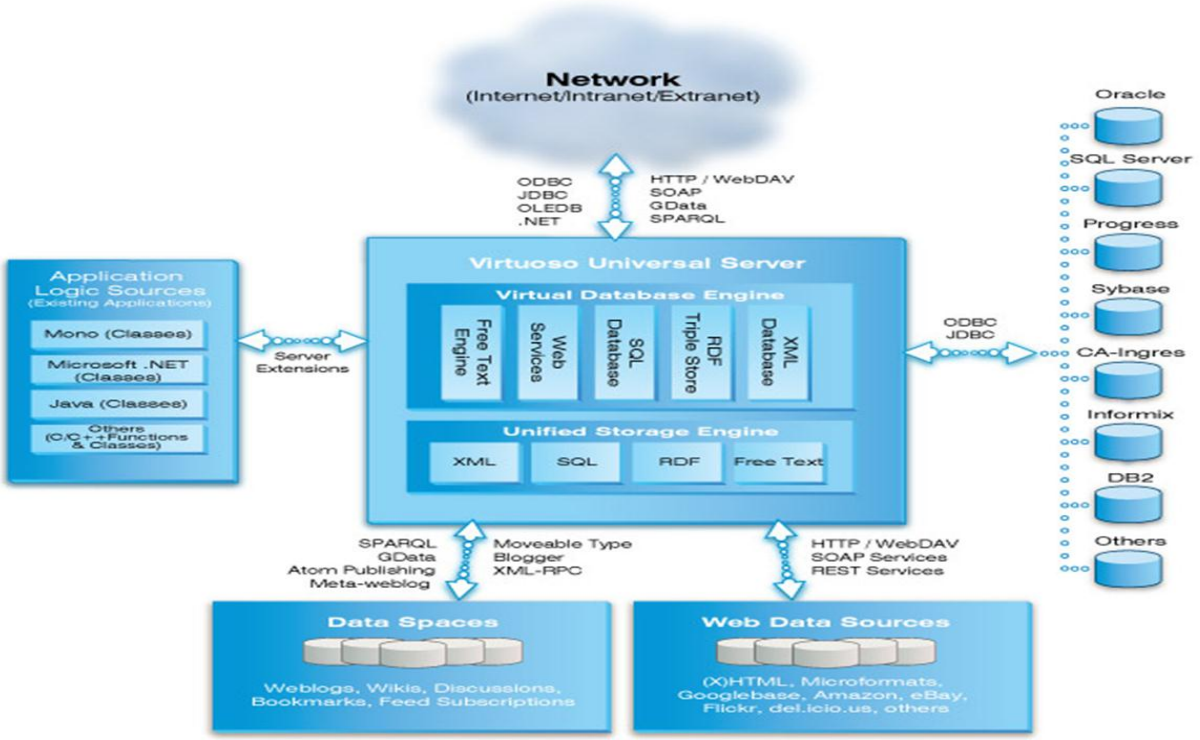


Figure 7-5: Virtuoso general architecture

8 Terms, Abbreviations and Definitions

Please add additional terms, abbreviations and definitions for your deliverable.

CRYSTAL	**CR**itical S**YST**em Enginieering **A**cce**L**eration
R	Report
P	Prototype
D	Demonstrator
O	Other
PU	Public
PP	Restricted to other program participants (including the JU).
RE	Restricted to a group specified by the consortium (including the JU).
CO	Confidential, only for members of the consortium (including the JU).
WP	Work Package
SP	Subproject

Table 8-1: Terms, Abbreviations and Definitions

9 Short literature review

Different frameworks such as The Ontolingua system or the Kactus project or the Protégé communities support a publicly accessible library of ontologies.

- OBO foundry
- http://protege.stanford.edu/
- http://hcs.science.uva.nl/projects/NewKACTUS/library/lib/
- A list of upper ontologies is provided at http://en.wikipedia.org/wiki/Upper_ontology
- http://www.omg.org/ontology/vocab_mgmt/

- **Naive Physics, fundational categories, and Commonsense Knowledge**
  - The aim of the CYC project (Lenat and Guha 1990) is to build up a large knowledge base with commonsense knowledge. To help structuring the knowledge in the knowledge base, an ontology of upper-level concepts has been developed.
  - Similarily, Dolce is the first module of the WonderWeb Foundational Ontologies Library (WFOL), introducing fundational categories such as *relations, functions, abstractions*, etc. http://www.loa.istc.cnr.it/DOLCE.html. A simplication is available at http://ontologydesignpatterns.org/wiki/Ontology:DOLCE+DnS_Ultralite
  - A review of formal ontologies for naïve physics before 1997 can be found in [Borst, 1997b].
  - Formalizations of knowledge about physical objects can be found in (Clarke 1981; Simons 1987; Cohn, Randell, and Cui 1995; Borgo, Guarino, and Masolo 1996b).
  - Quantities and Units ontologies can be found in OBO framework [REF], and OWL implementation of those in Dumontiers works for biology domain modeling
  - OGM Date Time vocabulary - http://www.omg.org/spec/DTV/1.0/
  - Quantities and Units of Measure (UoM) wiki http://ontolog.cim3.net/cgi-bin/wiki.pl?UoM
- **Engineering and Technical Applications**
  - Many ontologies have been developed for engineering and technical applications. An ontology for the Sisyphus elevator design problem (VT) is described in (Schreiber and Terpstra 1996).
  - In the KACTUS project (Laresgoiti, Anjewierden, Bernaras, Corera, Schreiber, and Wielinga 1996; Bernaras and Laresgoiti 1996), ontologies for diagnosis of electrical networks and for the exchange of knowledge about ship design and oil platforms have been written.

- The YMIR ontology (Alberts 1993) is a domain independent, sharable ontology for the formal representation of engineering design knowledge, based on systems theory.
- The PHYSSYS ontology (Borst, Akkermans, and Top 1997) has the same objectives. PHYSSYS is a formal ontology based upon system dynamics (Bond graphs). Knowledge formalized in PHYSSYS forms the basis for the OLMECO library, a model component library for physical systems like heating systems, automotive systems and machine tools.
- EngMath (Gruber 1994) is an ontology for mathematical modelling in engineering. It has been reused many times, for instance in PHYSSYS and in CML.
- Function ontologies were studied by Osaka in FOCUS project ("Functional Ontology for Categorization, Utilization and Systematization of functional knowledge") (http://www.ei.sanken.osaka-u.ac.jp/topics/Focus/fo-main.htm).
- Another work from National Institute of Standards and Technology [**Hirtz**] formalized a function related terminology reconcilitating previous independent research efforts from the NIST and other U.S. universities and their industrial partners.
- CML (Falkenhainer, Farquar, Bobrow, Fikes, Forbus, Gruber, Iwasaki, and Kuipers 1994) is an ontology about time, continuity, object properties etc. to enable the sharing of models based on compositional modelling (Falkenhainer and Forbus 1991; Forbus 1984). The CML ontology has been used to develop an ontology for thermodynamic systems and an ontology for VT.

- **Corporate Memory, Enterprise Modelling**
  - The (KA)2 ontologies (Decker et al., 1999) are examples of KM ontologies. They were built inside the Knowledge Annotation Initiative of the Knowledge Acquisition community, also known as the $(KA)^2$ initiative.
  - NEON Project objective is about using ontologies for large-scale semantic applications in the distributed organizations http://www.neon-project.org/nw/Welcome_to_the_NeOn_Project
  - The TOVE ontology (Fox, Chionglo, and Fadel 1993; Gruninger and Fox 1994), formalizes knowledge about production/ communication processes, activities, causality, resources, quality and cost in business enterprises.
  - Ontologies have also been developed for the implementation of knowledge bases for formalization and conservation of the knowledge of experts in enterprises. An example of such an ontology is the KONE ontology (Kühn 1994) that deals with conservation of corporate knowledge about crankshaft design.

10 References

[Author, Year]	Authors; *Title*; Publication data (document reference)
[Gruber, 1994a]	Gruber, T. R. *Towards principles for the design of ontologies used for knowledge sharing*. In N. Guarino and R. Poli (Eds.), *Formal Ontology in Conceptual Analysis and Knowledge Representation*. Kluwer. 1994
[Gruber, 1994b]	Gruber, T. R. *An Ontology for Engineering Mathematics*, In Jon Doyle, Piero Torasso, & Erik Sandewall, Eds., Fourth International Conference on Principles of Knowledge Representation and Reasoning, Gustav Stresemann Institut, Bonn, Germany, Morgan Kaufmann, 1994
[Guarino, 1997]	Guarino, N. *Some Organizing Principles For A Unified Top-Level Ontology*; 1997; Proceedings of the AAAI Spring Symposium on Ontological Engineering, Stanford (USA).
[Guarino, 1998]	Guarino, N. *Formal Ontology and Information Systems*; 1998; In N. Guarino (Eds), Formal Ontology in Information Systems, IOS Press, Amsterdam, pp. 3-15.
[Guarino & Welty, 2000]	Guarino, N. & Welty, C.; *A formal Ontology of Properties*; 2000; Proceedings of the ECAI-00 Workshop on Applications of Ontologies and Problem-Solving Methods, Berlin, pp. 12/1-12/8.
[Borst, 1997a]	Borst, Akkermans, Top, *Engineering ontologies* in Human Computer Studies 46 , 365 – 406, 1997
[Borst, 1997b]	W. N. Borst, Construction of engineering ontologies for knowledge sharing and reuse. CTIT Ph. D-thesis series No. 97-14, 1997.
[Bachimont, 2000]	Bachimont, B. *Engagement sémantique et engagement ontologique : conception et réalisation d'ontologies en ingénierie des connaissances* ; 2000; In J. Charlet et al. (eds), Ingénierie des Connaissances ; Evolutions récentes et nouveaux défis, Eyrolles, pp. 305-323.
[Gala, 2013]	Gala,N. Zock, M.. *Ressources Lexicales, Contenu, construction, utilisation, évaluation*. Lingvisticæ Investigationes Supplementa, 30, 2013, John Benjamin Publishing
OMG, ReqIF	Object Management Group. Oct 2013. Requirements Interchange Format (ReqIF); v.1.1. http://www.omg.org/spec/ReqIF/1.1/PDF/
OMG,	Documents Associated with Date-Time Vocabulary, V1.0
|Kyo, 1990]	Kyo C. Kang and al. Feature-Oriented Domain Analysis (FODA), 1990.
IWSSD '89	Proceedings of the 5th international workshop on Software specification and design 152-159. 1989

[Moros, 2008]	Begoña Moros, Cristina Vicente-Chicote, Ambrosio Toval, *Metamodeling Variability to Enable Requirements Reuse, in* Proceedings of EMMSAD 2008
Siegmund,	Norbert Siegmund, Martin Kuhlemann, Marko Rosenmuller, Christian Kaestner, and Gunter Saake, Integrated Product Line Model for Semi-Automated Product Derivation Using Non-Functional Properties, in
CRYSTAL D208.010	CRYSTAL aerospace use case description Report – V1 D208.010, 2014.
[CRYSTAL-D607-041]	CRYSTAL knowledgeMANAGER report – V1 D607.041, 2014.
[Lamm, 2013]	Lamm J.G. and Weilkiens, T.; Method for Deriving Functional Architectures from Use Cases, System Engineeering 21265, 2013.
[Chaudhry, 2010]	Chaudhry, A. S. (2010). Assessment of taxonomy building tools. The Electronic Library, 28(6), 769-788. doi:10.1108/02640471011093480
[Roszkiewicz, 2010]	Roszkiewicz, R. (2010). Enterprise metadata management: How consolidation simplifies control. Journal of Digital Asset Management, 6(5), 291-297. doi:10.1057/dam.2010.32
INCOSE DR1.2a	INCOSE JCAWG (Joint Commercial Aircraft Working Group): Framework for the Application of Systems Engineering in the Commercial Aircraft Domain - DRAFT – Version 1.2a – July 28, 2000
VIVACE, D1112.2	Virtual Aircraft Structure, Components And Scenarios, VIVACE, D.1.1.1.2, Issue 2.0. 2004.
Kitamura, 2006	Yoshinobu Kitamura*, Yusuke Koji and Riichiro Mizoguchi. An ontological model of device function:industrial deployment and lessons learned. *Applied Ontology, Vol. 1, No. 3-4, pp. 237-262, 2006.*
Hirtz, 2002	Julie Hirtz, Robert B. Stone, Daniel A. McAdams, Simon Szykman, and Kristin L. Wood. A Functional Basis for Engineering Design: Reconciling and Evolving Previous Efforts, NIST Technical Note 1447, 2002
Gkoutos	Georgios V. Gkoutos, Paul N. Schofield, and Robert Hoehndorf , The Units Ontology: a tool for integrating units of measurement in science, Database, Vol. 2012,
Kipper, 2006	Karin Kipper, Anna Korhonen, Neville Ryant, and Martha Palmer. Extensive Classifications of English verbs. Proceedings of the 12th EURALEX International Congress. Turin, Italy. September, 2006.
[Heiden 2010]	Heiden, S. (2010b). *The TXM Platform : Building Open-Source Textual Analysis Software Compatible with the TEI Encoding Scheme*. In K. I. Ryo Otoguro (Ed.), 24th Pacific Asia Conference on Language, Information and Computation - PACLIC24 (p. 389-398). Institute for Digital Enhancement of Cognitive Development, Waseda University, Sendai, Japan.

[MoSSEC 2013]	ASD STRATEGIC STANDARDISATION GROUP - MoSSEC Overview 2013 - ASD/MoSSEC/1.0

11 Annexes

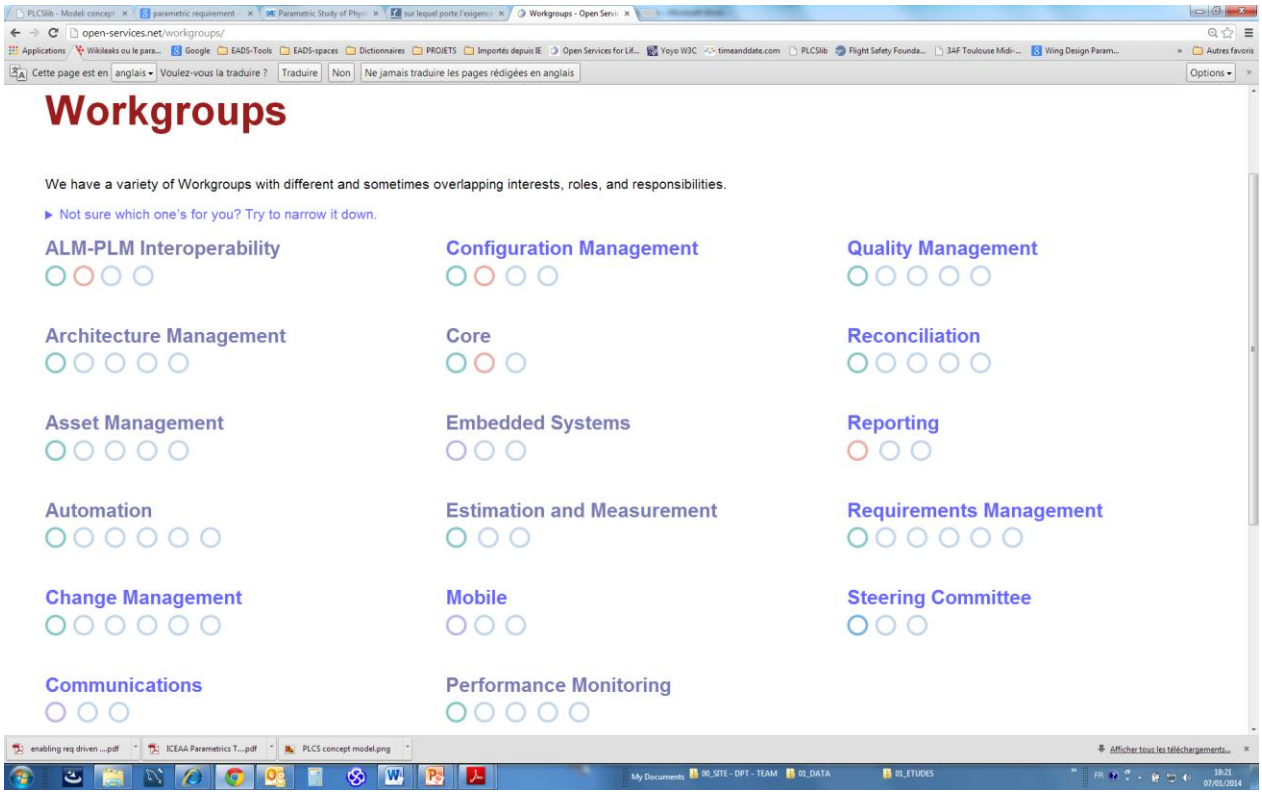
Annex I: DeX Specifications

The OASIS TC members have identified more than 30 candidate **DEX** specifications to meet particular industry needs. Here below a list of the available DES is provided

Product breakdown for support	Resource Schedule
Fault states	Resource Availability
Task set	Feedback Report
Work package definition	Activity Report
Maintenance Plan	Resource Report
Operational feedback	Work Justification
Product as individual	Impact Assessment
Work package reporting	Risk Assessment
System Requirements	Intended Product Use
Product Design	Actual Product Use
Part Definition information	Support Resources
Part with State	Support Personnel
Part as Individual	Support Organization
Part with Interface	Support Facility
Variant Specification	Allowance
Change Definition	Support Opportunity
Product Behavior	Product Behavior (Component)
Product Status Report	Product Behavior (Assembly)
Role Configuration Option	Resourced Task Specification
Work Management	Support Solution Justification
Work Request	Support Opportunity Types
Work Order	Actual Support Opportunity
Work Definition	

Annex II: OSLC concerns

The Open Services for Lifecycle Collaboration community has identified a set of concerns, the list of which can be retreived from http://open-services.net/. For most of them a Working group is working to define a common set of features that RESTful services should implement to enable concern's related activites.



Annex III: PLCS – an OWL map

Description: This ontology is a subset of PLCS ARM Long Form EXPRESS represented as OWL.

Version: Draft 5 – 09 September 2010

Copyright notice

Class **Activity** *[urn:plcs:rdl:std#Activity] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Activity is the identification of the occurrence of an action that has taken place, is taking place, or is expected to take place in the future. The procedure executed during that Activity is identified with the Activity_method that is referred to by the chosen_method attribute. EXAMPLE: Change, distilling, design, a process to drill a hole, and a task such as training someone, are examples of activities. NOTE: Status information identifying the level of completion of each activity may be provided within an instance of Activity_status. NOTE: The items that are affected by an Activity, for example as input or output, may be identified within an instance of Applied_activity_assignment.

```
CLASS Activity;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Activity_actual, Directed_activity, );
END_CLASS;
```

Class **Activity actual** *[urn:plcs:rdl:std#Activity_actual] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Activity_actual is a type of Activity. It is a record of the occurrence of an Activity.
The Activity_actual is related through an Activity_happening to the Activity for which it is an occurrence. The existence of an Activity_actual instance means that the Activity_actual has started. NOTE:
A Calendar_date or Date_time should be assigned to the Activity_actual with the role "start date" to record when the activity started. NOTE: A Calendar_date or Date_time may be assigned to the activity with role "end date" to record when the activity finished. In general, the absence of this assignment cannot be used to infer that the activity is continuing, only that the end of the activity has not yet been recorded. NOTE: A more detailed history of the progress of an activity may be recorded by applying states to the activity, but the meaning of these states must be defined through local business rules.

```
CLASS Activity_actual;
SUBCLASS OF ( Activity );
END_CLASS;
```

Class **Activity happening** *[urn:plcs:rdl:std#Activity_happening] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Activity_happening is a type of Activity_relationship. It is a relationship between the definition of an activity (Activity_happening.predicted) and its actual occurrence (Activity_happening.actual). NOTE: The ordinary value for Activity.name may be "actual", though this is redundant. NOTE:
Many Activity_actual s may be the actual for a single Activity. EXAMPLE: a single defined activity is recorded historically by several sub-activities. NOTE: A single Activity_actual may fulfil several activities. EXAMPLE: a single servicing activity takes the opportunity to make additional checks and repairs.

```
CLASS Activity_happening;
SUBCLASS OF ( Activity_relationship );
END_CLASS;
```

Class **Activity method** *[urn:plcs:rdl:std#Activity_method] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Activity_method is a way to carry out an Activity. NOTE: There may be more than one method for producing a required result NOTE: This definition may be used to characterize a way to resolve a request for action. The concept of action request is dealt with in module.

```
CLASS Activity_method;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Scheme, Task_method, Scheme_entry, Task_method_version,
Task_element, Scheme_version, );
END_CLASS;
```

Class **Activity method assignment** *[urn:plcs:rdl:std#Activity_method_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Activity_method_assignment is an association between an Activity_method and a Work_request. The relation_type attribute characterizes the meaning of that association and the meaning of the Activity_method with respect to the Work_request.

```
CLASS Activity_method_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Activity method realization** *[urn:plcs:rdl:std#Activity_method_realization] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Activity_method_realization is a further specification of how an activity method is to be performed with the Activity_method. NOTE: More than one specification can be associated with the same Activity_method. EXAMPLE: For a given planned activity there may be a task specification, a statement of how task performance is to be logged and a schedule that all apply.

```
CLASS Activity_method_realization;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Activity method realization relationship** *[urn:plcs:rdl:std#Activity_method_realization_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Activity_method_realization_relationship is a relationship between two Activity_method_realization entity instances. NOTE:
An Activity_method_realization_relationship may be used to specify sequencing and other constraints between different realizations for the same Activity_method.

```
CLASS Activity_method_realization_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Activity method relationship** *[urn:plcs:rdl:std#Activity_method_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Activity_method_relationship is a relationship between two instances of Activity_method.

```
CLASS Activity_method_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Scheme_version_relationship, Scheme_relationship,
Scheme_entry_relationship, Task_element_relationship,
Task_method_version_relationship, Task_method_relationship, );
END_CLASS;
```

Class **Activity property** *[urn:plcs:rdl:std#Activity_property] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Activity_property is a property of an Activity or of an Activity_method.

```
CLASS Activity_property;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Applied_independent_activity_property, );
END_CLASS;
```

Class **Activity property representation** *[urn:plcs:rdl:std#Activity_property_representation] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Activity_property_representation is an association between an Activity_property and one of its representations.

```
CLASS Activity_property_representation;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Activity relationship** *[urn:plcs:rdl:std#Activity_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Activity_relationship is a relationship between two instances of Activity. EXAMPLE: The activity required to complete a work order, may be decomposed into a series of activities. Their corresponding instances would be related using instances of the Activity_relationship entity.

```
CLASS Activity_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Activity_happening, );
END_CLASS;
```

Class **Activity status** *[urn:plcs:rdl:std#Activity_status] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Activity_status is the assignment of a status to an Activity.

```
CLASS Activity_status;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Address** *[urn:plcs:rdl:std#Address] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Address is the information that locate persons or organizations. It provides location details for communication via postal mail, telephone, facsimile, telex or electronic mail.

```
CLASS Address;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Address assignment** *[urn:plcs:rdl:std#Address_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Address_assignment is the association of an Organization or a Person_in_organization with an Address.

```
CLASS Address_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Address based location representation** *[urn:plcs:rdl:std#Address_based_location_representation] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Address_based_location_representation is a type of Location_representation specified by its postal identification.

```
CLASS Address_based_location_representation;
SUBCLASS OF ( Location_representation );
END_CLASS;
```

Class **Advisory task step** *[urn:plcs:rdl:std#Advisory_task_step] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Advisory_task_step is a type of Task_step. It conveys information. NOTE: Some Task_method s may not require any action to be undertaken. EXAMPLE: "Beware of hot exhausts", "do not use tool X this way" and similar messages.

```
CLASS Advisory_task_step;
SUBCLASS OF ( Task_step );
END_CLASS;
```

Class **Affected items assignment** *[urn:plcs:rdl:std#Affected_items_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Affected_items_assignment is an association of a Work_request with the product or activity data that are subjects of this Work_request. EXAMPLE: In case a tire on a car is flat, a Work_request may be created and associated with the instances that represent the tire that is flat, the car and the spare wheel.

```
CLASS Affected_items_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Alias identification** *[urn:plcs:rdl:std#Alias_identification] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Alias_identification is a type of Identification_assignment that provides an alias identifier to product or activity data. NOTE: The alias identifier is conveyed within the identifier attribute inherited from Identification_assignment. NOTE: Information about the organizational scope in which the alias applies may be provided by a specialization of Organization_or_person_in_organization_assignment. NOTE: Alias identification only applies to concepts that possesses an attribute that conveys an identifier.

```
CLASS Alias_identification;
SUBCLASS OF ( Identification_assignment );
END_CLASS;
```

Class **Alternate part relationship** *[urn:plcs:rdl:std#Alternate_part_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Alternate_part_relationship is a type of alternate_product_relationship where the alternate and base products are parts.

```
CLASS Alternate_part_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Amount of substance unit** *[urn:plcs:rdl:std#Amount_of_substance_unit] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Amount_of_substance_unit is a type of Unit in which the number of elementary entities of a substance as compared to the number of atoms in 0.012 kilograms of carbon-12 is expressed. NOTE: This definition applies to the SI quantity 'mole'. When the mole is used, the elementary entities, whose quantity is expressed, must be specified . They may be atoms, molecules, ions, electrons or other particles or specified groups of such particles (see ISO 31-8). NOTE: This unit corresponds to one of the seven fundamental quantities as specified in ISO 1000.

```
CLASS Amount_of_substance_unit;
SUBCLASS OF ( Unit );
END_CLASS;
```

Class **And state cause effect definition** *[urn:plcs:rdl:std#And_state_cause_effect_definition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An And_state_cause_effect_definition is a type of State_cause_effect_definition. It relates one or more causing State_definition entities and one effect State_definition. All the causing State_definition entities must exist prior to the single effect.

```
CLASS And_state_cause_effect_definition;
SUBCLASS OF ( State_cause_effect_definition );
END_CLASS;
```

Class **Applied activity assignment** *[urn:plcs:rdl:std#Applied_activity_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Applied_activity_assignment is an association of an Activity with product or activity data. It characterizes the role of the concepts represented with these data with respect to the activity. NOTE: This entity should not be used to represent the association of an activity with the organizations that are responsible for its execution or its management. That kind of information can be represented with instances of Organization_or_person_in_organization_assignment.

```
CLASS Applied_activity_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Applied activity method assignment** *[urn:plcs:rdl:std#Applied_activity_method_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Applied_activity_method_assignment is an association of an Activity_method with product or activity data.

```
CLASS Applied_activity_method_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Task_element_assignment, Scheme_subject_assignment,
Scheme_entry_assignment, Task_method_version_assignment, Task_method_assignment,
Scheme_version_assignment, );
END_CLASS;
```

Class **Applied independent activity property** *[urn:plcs:rdl:std#Applied_independent_activity_property] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Applied_independent_activity_property is a type of Activity_property that is an association of an Activity_property with an Independent_property. It characterizes the fact that the Activity_property is the application of the Independent_property to activity data.

```
CLASS Applied_independent_activity_property;
SUBCLASS OF ( Activity_property );
```

END_CLASS;

Class **Applied independent property** *[urn:plcs:rdl:std#Applied_independent_property] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Applied_independent_property is a type of Assigned_property that is associated with an Independent_property. It characterizes the fact that the Applied_independent_property is the application of the Independent_property to product data.

```
CLASS Applied_independent_property;
SUBCLASS OF ( Assigned_property );
END_CLASS;
```

Class **Applied independent resource property** *[urn:plcs:rdl:std#Applied_independent_resource_property] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Applied_independent_resource_property is a type of Resource_property that is associated to a Independent_property. It characterizes the fact that the Resource_property is the application of the Independent_property to activity data.

```
CLASS Applied_independent_resource_property;
SUBCLASS OF ( Resource_property );
END_CLASS;
```

Class **Applied information usage right** *[urn:plcs:rdl:std#Applied_information_usage_right] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Applied_information_usage_right is an application of a particular usage right to a set of items. NOTE: If an approval is applied to this entity, it carries the meaning that the particular set of items is approved for the given usage. This approval generally indicates that the approval is exceptional, for example, where the information belongs to another project, and that project agrees to share some particular items of information. There is a further implication that the set of entities should not be changed once the approval is given.

```
CLASS Applied_information_usage_right;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Applied state assignment** *[urn:plcs:rdl:std#Applied_state_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Applied_state_assignment is a relationship that allows a subject to have State or to be in a State.

```
CLASS Applied_state_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Applied state definition assignment**
[urn:plcs:rdl:std#Applied_state_definition_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Applied_state_definition_assignment is a mechanism that enables an object to have or to be in a State_definition.

```
CLASS Applied_state_definition_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Approval** *[urn:plcs:rdl:std#Approval] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Approval is a formal confirmation of the quality of some activity or product data.

```
CLASS Approval;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Approval assignment** *[urn:plcs:rdl:std#Approval_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Approval_assignment is the assignment of an Approval to activity or product data.

```
CLASS Approval_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Approval relationship** *[urn:plcs:rdl:std#Approval_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Approval_relationship is a typed association between two instances of Approval.

```
CLASS Approval_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Approval status** *[urn:plcs:rdl:std#Approval_status] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Approval_status is a particular rank of approval.

```
CLASS Approval_status;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Approving person organization** *[urn:plcs:rdl:std#Approving_person_organization] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Approving_person_organization is an association between an Approval and the organization or person and organization that has granted this approval.

```
CLASS Approving_person_organization;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Assembly component relationship** *[urn:plcs:rdl:std#Assembly_component_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Assembly_component_relationship is a type of View_definition_usage. It identifies a possibly quantified usage of a product version as a component of another product version. The relationship is established between two instances of Product_view_definition. The inherited attribute View_definition_relationship.relating_view identifies the Product_view_definition of the assembly. The inherited attribute View_definition_relationship.related_view identifies the Product_view_definition of the product version that plays the role of component. The Assembly_component_relationship specifies that, in the Product_view_definition.initial_context of the Product_view_definition that is referred to as View_definition_relationship.relating_view, it is considered that the product version that is indirectly identified with the View_definition_relationship.related_view attribute, is a component of the product version that is indirectly identified with the View_definition_relationship.relating_view attribute. NOTE: In another context, the structure of the assembly may be described differently, adding, for example, an intermediate level between the products. NOTE: This entity data type may be used to establish assembly relationships during design or to represent the composition of an assembly existing in the real world. NOTE: An Assembly_component_relationship identifies an item in a parts list. Should the quantity be zero, the component would still be listed in the parts list. In case the component is a part, the following additional specifications apply: * the quantity attribute shall either be not specified or shall characterize a number of occurrences of the component; * if the quantity attribute specifies a number of occurrences, these occurrences shall be considered as a single group within the assembly structure; * if the quantity attribute is not specified, the relationship actually identifies a single occurrence of the component. In case the component is a non-countable material, the following additional specifications apply: * the quantity attribute shall either be not specified or shall characterize the amount of the material used as component; * if the quantity attribute is not specified, the amount of material used as component shall be considered as unknown. EXAMPLE: An assembly may require inclusion of ten grams of grease. NOTE: This version of the Assembly structure module does not enable to represent the fact that the quantity of a fluid component is, for example 'at most 10 grams' or 'between 5 or 20 grams'. However, some ISO 10303 application protocols, for example ISO 10303-214, provide a representation for those requirements. An Assembly_component_relationship shall be either a Next_assembly_usage, a or a Component_upper_level_identification.

```
CLASS Assembly_component_relationship;
SUBCLASS OF ( View_definition_usage );
SUPERCLASS OF ( Component_upper_level_identification, Promissory_usage,
Next_assembly_usage, );
END_CLASS;
```

Class **Assembly relationship substitution** *[urn:plcs:rdl:std#Assembly_relationship_substitution] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Assembly_relationship_substitution is a relationship that indicates that an Assembly_component_relationship may be substituted by another Assembly_component_relationship. Both assembly relationships shall refer to the same Product_view_definition of the same assembly. NOTE: Consequently, an Assembly_relationship_substitution actually specifies that the product version that plays the role of component in the substitute_relationship may be replaced by the product version that plays the role of component in the base_relationship. NOTE: The instance of the substitute constituent does not require the same spatial relationship or the same quantity. A substitute constituent does not require equivalent form, fit, and function of the constituent for which it is a substitute. NOTE: As instances of Assembly_component_relationship establish assembly relationships relevant in the definition contexts of the assembly, the substitution only apply in these contexts. An Assembly_relationship_substitution defines a one-way substitution: if A is specified as a substitute for B, B is not implied to be a substitute for A.

```
CLASS Assembly_relationship_substitution;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Assigned document property** *[urn:plcs:rdl:std#Assigned_document_property] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Assigned_document_property is a type of Assigned_property that identifies a property assigned to a document definition or to a file. When the Assigned_document_property is applied to a Digital_document_definition through the Assigned_property.described_element attribute, the Assigned_document_property applies to all instances of Digital_file that are components of the Digital_document_definition. When the Assigned_document_property is applied to a Physical_document_definition through the Assigned_property.described_element attribute, the Assigned_document_property applies to all instances of Hardcopy that are components of the Physical_document_definition. When the Assigned_document_property is applied to a File through the Assigned_property.described_element attribute, the Assigned_document_property applies to the individual File.

```
CLASS Assigned_document_property;
SUBCLASS OF ( Assigned_property );
END_CLASS;
```

Class **Assigned property** *[urn:plcs:rdl:std#Assigned_property] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Assigned_property is a property that is assigned to product data. NOTE: The mapping provided for that entity and for its attribute described_element is incomplete. It needs to be completed in any module that uses this module and extends the type property_assignment_select.

```
CLASS Assigned_property;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Assigned_document_property, Applied_independent_property, );
END_CLASS;
```

Class **Attachment slot** *[urn:plcs:rdl:std#Attachment_slot] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Attachment_slot is a type of Product that represents the position in which a part is or can be attached to a parent product. EXAMPLE: A fast jet aircraft has two engines. These engines are removable and interchangeable between individual aircraft. An attachment slot represents each installation position for an engine so as to ensure that an accurate record is maintained of which engines fly in which pairing on which aircraft for how many hours.

```
CLASS Attachment_slot;
SUBCLASS OF ( Product );
END_CLASS;
```

Class **Attachment slot as planned** *[urn:plcs:rdl:std#Attachment_slot_as_planned] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Attachment_slot_as_planned is a type of Attachment_slot_version that identifies an individual that is the subject of a plan to realize an Attachment_slot. EXAMPLE: FlyFasterWithUs Group will buy an aircraft with serial number 1234 next year. The company wishes to plan the schedule for removal of engines from the aircraft for maintenance purposes. Instances of the Attachment_slot_as_planned entity data type allow the company to associate individual engines with the aircraft at different times over the planned period.

```
CLASS Attachment_slot_as_planned;
SUBCLASS OF ( Attachment_slot_version );
END_CLASS;
```

Class **Attachment slot as realized** *[urn:plcs:rdl:std#Attachment_slot_as_realized] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Attachment_slot_as_realized is a type of Attachment_slot_version that identifies an individual that is a realized Attachment_slot. EXAMPLE: FlyFasterWithUs Group operates an aircraft with serial number 1234 next year. The company records which individual engines power the aircraft at different times during the lifetime of the aircraft.

```
CLASS Attachment_slot_as_realized;
SUBCLASS OF ( Attachment_slot_version );
END_CLASS;
```

Class **Attachment slot definition** *[urn:plcs:rdl:std#Attachment_slot_definition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Attachment_slot_definition is a type of Product_view_definition that identifies a view of an Attachment_slot. EXAMPLE: An airworthiness authority requires an airline company to report which individual engines power the aircraft at different times during the lifetime of the aircraft.

```
CLASS Attachment_slot_definition;
SUBCLASS OF ( Product_view_definition );
```

END_CLASS;

Class **Attachment slot design** *[urn:plcs:rdl:std#Attachment_slot_design] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Attachment_slot_design is a type of Attachment_slot_version that identifies the design version of an attachment slot. EXAMPLE: WeMakeBigPlanes Limited creates design version 1.34 of the attachment slot for the starboard engine of an aircraft.

CLASS **Attachment_slot_design**;
SUBCLASS OF (Attachment_slot_version);
END_CLASS;

Class **Attachment slot design to planned** *[urn:plcs:rdl:std#Attachment_slot_design_to_planned] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Attachment_slot_design_to_planned is a relationship between a design version of an Attachment_slot and a planned individual that conforms to the design. EXAMPLE: WeMakeBigPlanes Limited plans production of aircraft serial number 1234 with a starboard engine attachment slot that is to conform to design version 1.34.

CLASS **Attachment_slot_design_to_planned**;
SUBCLASS OF (PLCS-ARM-LF-THING);
END_CLASS;

Class **Attachment slot design to realized** *[urn:plcs:rdl:std#Attachment_slot_design_to_realized] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Attachment_slot_design_to_realized is a relationship between a design version of an Attachment_slot and a realized individual that conforms to the design. EXAMPLE: WeMakeBigPlanes Limited builds aircraft serial number 1234 with a starboard engine attachment slot that conforms to design version 1.34.

CLASS **Attachment_slot_design_to_realized**;
SUBCLASS OF (PLCS-ARM-LF-THING);
END_CLASS;

Class **Attachment slot on product** *[urn:plcs:rdl:std#Attachment_slot_on_product] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Attachment_slot_on_product is a relationship between a product and an Attachment_slot that is a location on the product at which to install removable parts. EXAMPLE: An aircraft has a pylon mounting on a wing as a location at which to install various equipment. An instance of the Attachment_slot_on_product entity data type identifies which attachment slot corresponds to the pylon.

CLASS **Attachment_slot_on_product**;
SUBCLASS OF (PLCS-ARM-LF-THING);
END_CLASS;

Class **Attachment slot planned to realized**
[urn:plcs:rdl:std#Attachment_slot_planned_to_realized] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Attachment_slot_planned_to_realized is a relationship between a realized individual of an Attachment_slot and a corresponding planned individual. EXAMPLE: WeMakeBigPlanes Limited builds aircraft serial number 2468 with a starboard engine attachment slot that was previously planned.

```
CLASS Attachment_slot_planned_to_realized;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Attachment slot version** *[urn:plcs:rdl:std#Attachment_slot_version] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Attachment_slot_version is a type of Product_version that identifies a version of Attachment_slot. NOTE: This is a generic concept of version, in most situations it is possible and more specific to represent a version as Attachment_slot_design, Attachment_slot_as_planned or Attachment_slot_as_realized.

```
CLASS Attachment_slot_version;
SUBCLASS OF ( Product_version );
SUPERCLASS OF ( Attachment_slot_design, Attachment_slot_as_realized,
Attachment_slot_as_planned, );
END_CLASS;
```

Class **Attribute translation assignment** *[urn:plcs:rdl:std#Attribute_translation_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Attribute_translation_assignment is the assignment of a translation to a text attribute of an instance.

```
CLASS Attribute_translation_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Axis placement** *[urn:plcs:rdl:std#Axis_placement] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Axis_placement is a type of Detailed_geometric_model_element that defines a right-handed, 2D or 3D, coordinate system. If the Axis_placement belongs to a 3D geometric space, the third direction of the coordinate system is defined by the vector product of x-axis and y-axis.

```
CLASS Axis_placement;
SUBCLASS OF ( Detailed_geometric_model_element );
END_CLASS;
```

Class **Breakdown** *[urn:plcs:rdl:std#Breakdown] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Breakdown is a type of Product that identifies a partitioning of a product into a set of related elements so as to form explicit, parent-child views that comprise the product elements. The parent-child view is represented by Breakdown_element_usage objects relating the elements in the breakdown that are represented by Breakdown_element objects. A Breakdown may be: * a Functional_breakdown, * a Hybrid_breakdown, * a Physical_breakdown, * a System_breakdown, or * a Zone_breakdown.

```
CLASS Breakdown;
SUBCLASS OF ( Product );
SUPERCLASS OF ( Hybrid_breakdown, Zone_breakdown, System_breakdown,
Functional_breakdown, Physical_breakdown, );
END_CLASS;
```

Class **Breakdown context** *[urn:plcs:rdl:std#Breakdown_context] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Breakdown_context is a membership relationship between a Breakdown_element and a Breakdown of which the element is a member.

```
CLASS Breakdown_context;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Physical_breakdown_context, Functional_breakdown_context,
Hybrid_breakdown_context, System_breakdown_context, Zone_breakdown_context, );
END_CLASS;
```

Class **Breakdown element** *[urn:plcs:rdl:std#Breakdown_element] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Breakdown_element is a type of Product that identifies the elements in one or more Breakdown objects. NOTE: Breakdown_element is non-specific and allows for various types of product breakdown. The more specific breakdown elements are Functional_element, Physical_element, System_element, and Zone_element.

```
CLASS Breakdown_element;
SUBCLASS OF ( Product );
SUPERCLASS OF ( Zone_element, System_element, Functional_element,
Physical_element, );
END_CLASS;
```

Class **Breakdown element definition** *[urn:plcs:rdl:std#Breakdown_element_definition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Breakdown_element_definition is a type of Product_view_definition that identifies a view of a version (Breakdown_element_version) of a Breakdown_element. NOTE: Breakdown_element_definition is non-specific and allows for various types of product breakdowns. The more specific breakdown element definitions

are Functional_element_definition, Physical_element_definition, System_element_definition and Zone_element_definition.

```
CLASS Breakdown_element_definition;
SUBCLASS OF ( Product_view_definition );
SUPERCLASS OF ( Functional_element_definition, System_element_definition,
Zone_element_definition, Physical_element_definition, );
END_CLASS;
```

Class **Breakdown element realization** *[urn:plcs:rdl:std#Breakdown_element_realization] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Breakdown_element_realization is a type of Product_definition_element_relationship that identifies a relationship between a Breakdown_element_definition or a Breakdown_element_usage and an item that realizes that element definition or usage. EXAMPLE: A pump realizes the 'provide fuel to engine' element in a functional breakdown for a ship.

```
CLASS Breakdown_element_realization;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Breakdown element usage** *[urn:plcs:rdl:std#Breakdown_element_usage] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Breakdown_element_usage is a type of View_definition_relationship that identifies a relationship between a parent and child Breakdown_element.

```
CLASS Breakdown_element_usage;
SUBCLASS OF ( View_definition_usage );
SUPERCLASS OF ( Physical_element_usage, System_element_usage,
Zone_element_usage, Hybrid_element_usage, Functional_element_usage, );
END_CLASS;
```

Class **Breakdown element version** *[urn:plcs:rdl:std#Breakdown_element_version] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Breakdown_element_version is a type of Product_version that identifies a version of a Breakdown_element.. NOTE: Breakdown_element_version is non-specific and allows for various types of product breakdown. The more specific breakdown elements are Functional_element_version, Physical_element_version, System_element_version and Zone_element_version.

```
CLASS Breakdown_element_version;
SUBCLASS OF ( Product_version );
SUPERCLASS OF ( Zone_element_version, Functional_element_version,
System_element_version, Physical_element_version, );
END_CLASS;
```

Class **Breakdown of** *[urn:plcs:rdl:std#Breakdown_of] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Breakdown_of is a relationship between a Breakdown and a Product of which the breakdown is a view.

```
CLASS Breakdown_of;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Breakdown version** *[urn:plcs:rdl:std#Breakdown_version] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Breakdown_version is a type of Product_version that identifies a version of a Breakdown.

```
CLASS Breakdown_version;
SUBCLASS OF ( Product_version );
SUPERCLASS OF ( Physical_breakdown_version, Functional_breakdown_version,
Hybrid_breakdown_version, System_breakdown_version, Zone_breakdown_version, );
END_CLASS;
```

Class **Calendar date** *[urn:plcs:rdl:std#Calendar_date] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Calendar_date is a date that is defined as a day in a month of a year.

```
CLASS Calendar_date;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Cartesian point** *[urn:plcs:rdl:std#Cartesian_point] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Cartesian_point is a type of Detailed_geometric_model_element that defines a point by a list of up to 3 cartesian coordinates.

```
CLASS Cartesian_point;
SUBCLASS OF ( Detailed_geometric_model_element );
END_CLASS;
```

Class **Cartesian transformation 2d** *[urn:plcs:rdl:std#Cartesian_transformation_2d] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Cartesian_transformation_2d is a type of Geometric_model. It is defined in a 2D geometric space by a 2*2 matrix and a cartesian point. Let be: * M, the 2*2 multiplication matrix of the cartesian transformation; * A, the point of the cartesian transformation; * P, a point in the geometric space; * Q, the result of the application of the transformation to P. The coordinates of Q shall be obtained by the formula: Q = M*P + A

```
CLASS Cartesian_transformation_2d;
```

SUBCLASS OF (Detailed_geometric_model_element);
END_CLASS;

Class **Cartesian transformation 3d** *[urn:plcs:rdl:std#Cartesian_transformation_3d] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Cartesian_transformation_3d is a type of Detailed_geometric_model_element that is a geometric transformation defined in a 3D geometric space by a 3*3 matrix and a cartesian point. Let be: * M, the 3*3 multiplication matrix of the cartesian transformation; * A, the point of the cartesian transformation; * P, a point in the geometric space; * Q, the result of the application of the transformation to P. The coordinates of Q shall be obtained by the formula: Q = M*P + A

CLASS **Cartesian_transformation_3d;**
SUBCLASS OF (Detailed_geometric_model_element);
END_CLASS;

Class **Certification** *[urn:plcs:rdl:std#Certification] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Certification is a certificate. It asserts satisfaction of particular quality criteria. NOTE: Certification information can be attached to any aspect of product or activity data.

CLASS **Certification;**
SUBCLASS OF (PLCS-ARM-LF-THING);
END_CLASS;

Class **Certification assignment** *[urn:plcs:rdl:std#Certification_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Certification_assignment is an association of a Certification with activity or product data. EXAMPLE: 'certified supplier' is an example of certification that may be granted to an organization by its contractor.

CLASS **Certification_assignment;**
SUBCLASS OF (PLCS-ARM-LF-THING);
END_CLASS;

Class **Characterizable object** *[urn:plcs:rdl:std#Characterizable_object] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Characterizable_object is an object that may be characterized with properties but that cannot be considered as a product or an activity. EXAMPLE: An orbit of a satellite is an example of Characterizable_object. NOTE: A Characterizable_object may be used as an environmental condition under which the properties of a product are measured. EXAMPLE: If a product has a set of properties that are measured within a particular atmosphere, the atmosphere may be described with an instance of the entity data type Characterizable_object.

CLASS **Characterizable_object;**
SUBCLASS OF (PLCS-ARM-LF-THING);
END_CLASS;

Class **Class** *[urn:plcs:rdl:std#Class] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Class is a number of things considered together. NOTE: In this part of ISO 10303, the term 'class' and 'set' are synonyms. NOTE: A class can consist of all things with a particular set of properties. Hence information about the consequences of possessing the set of properties can be assigned to the class. If a thing is classified as being a member of such a class, then a set of properties possessed by the thing can be deduced. NOTE: This entity may be instantiated as a compound instance involving another entity from ISO 10303. Each Class is a Class_by_extension or a Class_by_intension NOTE: The distinction between a Class_by_extension and a Class_by_intension can be imprecise. For example, the set of items produced by a particular production run could be regarded as either. The entity type class is not specified as abstract, so an application protocol or application module can decide to ignore the distinction.

```
CLASS Class;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Class_by_intension, Class_by_extension, External_class,
Selected_item, );
END_CLASS;
```

Class **Class by extension** *[urn:plcs:rdl:std#Class_by_extension] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Class_by_extension is a type of Class that is a set that is defined by means of a criterion that does not enumerate the members.

```
CLASS Class_by_extension;
SUBCLASS OF ( Class );
END_CLASS;
```

Class **Class by intension** *[urn:plcs:rdl:std#Class_by_intension] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Class_by_intension is a type of Class that is a set that is defined by enumerating the members.

```
CLASS Class_by_intension;
SUBCLASS OF ( Class );
END_CLASS;
```

Class **Component upper level identification** *[urn:plcs:rdl:std#Component_upper_level_identification] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Component_upper_level_identification is a type of Assembly_component_relationship. It identifies a component of an assembly with respect to an upper level in the assembly structure. NOTE:
A Component_upper_level_identification does not add a component in an assembly, it provides a means to further characterize a component with respect to an upper level assembly. EXAMPLE:
A Component_upper_level_identification may be used to assign a property to a component that applies in the context of a particular upper level assembly. The identified component is the version of product,

indirectly referred to as the View_definition_relationship.related_view of the sub_assembly_relationship. The upper level assembly is the version of product, indirectly referred to as the View_definition_relationship.relating_view of the upper_assembly_relationship.

```
CLASS Component_upper_level_identification;
SUBCLASS OF ( Assembly_component_relationship );
END_CLASS;
```

Class **Composition of state** *[urn:plcs:rdl:std#Composition_of_state] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Composition_of_state is a type of state relationship and it relates the nature of states in relation to one another, where two or more State parts compose a whole State, and furthermore, whole states can become parts of yet another whole State.

```
CLASS Composition_of_state;
SUBCLASS OF ( State_relationship );
END_CLASS;
```

Class **Composition of state definition** *[urn:plcs:rdl:std#Composition_of_state_definition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Composition_of_state_definition is a type of State_definition_relationship. It relates State_definition entities to one another, when two or more State_definition entities act as parts to compose a whole State_definition, and furthermore, whole State_definition entities can become parts of yet another whole State_definition.

```
CLASS Composition_of_state_definition;
SUBCLASS OF ( State_definition_relationship );
END_CLASS;
```

Class **Concurrent elements** *[urn:plcs:rdl:std#Concurrent_elements] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Concurrent_elements is a type of Structured_task_element that comprises a set of actions to be performed during the time required for the longest task. No specific order is required.

```
CLASS Concurrent_elements;
SUBCLASS OF ( Structured_task_element );
SUPERCLASS OF ( Simultaneous_elements, );
END_CLASS;
```

Class **Condition** *[urn:plcs:rdl:std#Condition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Condition is a definition of the precedent that must be fulfilled before a statement or relationship becomes valid. NOTE: The condition is defined as a text based expression that is represented by the description attribute. The parameters against which the condition is to be evaluated are identified by Condition_parameter. The target or consequence of a condition is represented

by Condition_assignment. EXAMPLE: "If the engine has been running for 10000 hours then it requires a service" is an example of a conditional statement. The conditional part of the statement is "If the engine has been running for 10000 hours" which is stored in definition attribute on Condition. The parameter or subject of the condition is "the engine" which is represented by a Condition_parameter identifying the Product_as_realized which represents the engine. The consequence of the condition is "then it requires a service". This is represented by Condition_assignment identifying the task to perform the service, a Task_method.

```
CLASS Condition;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Condition assignment** *[urn:plcs:rdl:std#Condition_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Condition_assignment is a relationship that identifies the statement or relationship to which a Condition applies. EXAMPLE: Condition 29 applies to the relationship between a Saab 9.3 car and the activity of checking the oil level on that make of car.

```
CLASS Condition_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Condition evaluation** *[urn:plcs:rdl:std#Condition_evaluation] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Condition_evaluation is a record of the evaluation of a Condition and the subsequent result. EXAMPLE: A Condition is "If the measured value of oil pressure from gauge 3 on a car is less than 2 bar then check the oil level" When the condition is evaluated it is recorded by an instance of Condition_evaluation. The measured value of oil pressure from gauge 3 on car with VIN 12345678 is 1.9 bar. Therefore the result of the evaluated the condition is true.

```
CLASS Condition_evaluation;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Condition evaluation assignment** *[urn:plcs:rdl:std#Condition_evaluation_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Condition_evaluation_assignment is a relationship that identifies the statement or relationship to which the Condition_evaluation applies. EXAMPLE: The Condition_evaluation (instance 87) is assigned to the activity of checking the oil level on car VIN 12345678.

```
CLASS Condition_evaluation_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Condition evaluation parameter** *[urn:plcs:rdl:std#Condition_evaluation_parameter] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Condition_evaluation_parameter is an identification of the product or activity data used in the evaluation of the Condition identified by the Condition_evaluation. EXAMPLE: The measured value of oil pressure from gauge 3 on car with VIN 12345678 (value = 1.9 bar). NOTE: The product or activity data is defined in condition_evaluation_parameter_item. The contents of this select type are defined in application modules that use this module.

```
CLASS Condition_evaluation_parameter;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Condition parameter** *[urn:plcs:rdl:std#Condition_parameter] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Condition_parameter is a represention of the product or activity data that is used to specify a Condition. EXAMPLE: Oil pressure on gauge 3. NOTE: The product or activity data is defined in condition_parameter_item. The contents of this select type are defined in application modules that use this module.

```
CLASS Condition_parameter;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Condition relationship** *[urn:plcs:rdl:std#Condition_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Condition_relationship is a relation between two conditions. NOTE: The Condition_relationship normally represents a logical combination of conditions. The logical type is identified by the classification of the Condition_relationship by a Classification_assignment. EXAMPLE: "If the engine has been running for 10000 hours AND the engine is fitted with a clog-up-quick Oil filter then change the oil filter" is an example of two conditions related by a logical AND.

```
CLASS Condition_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Content item** *[urn:plcs:rdl:std#Content_item] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Content_item is a reference to any item that can be referenced as part of the message content. NOTE: As well as externally defined items such as drawings, photographs, and so on, a message can refer to any entity represented in the schema it is used in, from products to property values, or actions to task steps. The mechanism used gives no interpretation of how the Content_item.item_identifier is interpreted or how the data is accessed. For example, the identifier could be the number of the entity in a file that conforms to ISO 10303-21. Interpretation of this entity and its attributes may be defined through reference data.

```
CLASS Content_item;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Context dependent unit** *[urn:plcs:rdl:std#Context_dependent_unit] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Context_dependent_unit is a type of Unit that is not related to the system of units defined in this part of ISO 10303. EXAMPLE: The number of parts in an assembly is a physical quantity that may be measured in a unit called 'parts'. Such a unit cannot be related to an SI unit.

```
CLASS Context_dependent_unit;
SUBCLASS OF ( Unit );
END_CLASS;
```

Class **Contextual item shape** *[urn:plcs:rdl:std#Contextual_item_shape] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Contextual_item_shape is a type of Item_shape that identifies the shape of a product version in the context of its use in another product version. The product version whose contextual shape is identified, is the product version associated with the View_definition_relationship.related_view of the View_definition_usage. EXAMPLE: Flexible part may have several shapes, each associated with a particular occurrence of the part in assemblies.

```
CLASS Contextual_item_shape;
SUBCLASS OF ( Item_shape );
END_CLASS;
```

Class **Contract** *[urn:plcs:rdl:std#Contract] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Contract is a binding agreement.

```
CLASS Contract;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Contract assignment** *[urn:plcs:rdl:std#Contract_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Contract_assignment is an association of a Contract with activity or product data.

```
CLASS Contract_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Conversion based unit** *[urn:plcs:rdl:std#Conversion_based_unit] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Conversion_based_unit is a type of Unit that that is based on another and related by a conversion factor. NOTE: A Conversion_based_unit is often used to convert a unit in one system of units to a similar unit in another system. EXAMPLE: An Imperial inch is 25.4 millimetres

```
CLASS Conversion_based_unit;
SUBCLASS OF ( Unit );
END_CLASS;
```

Class **Date or date time assignment** *[urn:plcs:rdl:std#Date_or_date_time_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Date_or_date_time_assignment is an association of a Calendar_date or a Date_time with activity or product data.

```
CLASS Date_or_date_time_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Date time** *[urn:plcs:rdl:std#Date_time] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Date_time is a time on a particular day.

```
CLASS Date_time;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Dated effectivity** *[urn:plcs:rdl:std#Dated_effectivity] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Dated_effectivity is a type of Effectivity for which the domain of applicability is defined as an interval of time bounded by dates or events. The interval may be open-ended. EXAMPLE: Events may be used to bound a Dated_effectivity period, at planning phase. Depending on whether the Dated_effectivity.end_bound attribute is specified, the actual domain of time defined by a Dated_effectivity is: * either, the time interval between the start and end date or event; * or, the open time interval that starts at the start date or event. If the Dated_effectivity.end_bound is an event that actually identifies a point in time that comes before the Dated_effectivity.start_bound, then the actual domain of effectivity is empty.

```
CLASS Dated_effectivity;
SUBCLASS OF ( Effectivity );
END_CLASS;
```

Class **Decision point** *[urn:plcs:rdl:std#Decision_point] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Decision_point is a type of Structured_task_element. It requires a decision that determines which further Task_element is to be followed. NOTE: The condition on which the decision is based optionally allows a further Task_element to be invoked to provide the basis for the decision.

```
CLASS Decision_point;
SUBCLASS OF ( Structured_task_element );
END_CLASS;
```

Class **Decreasing resource event** *[urn:plcs:rdl:std#Decreasing_resource_event] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Decreasing_resource_event is a type of Resource_event that decreases the balance of a managed resource. EXAMPLE: Filling a requisition reduces an inventory.

```
CLASS Decreasing_resource_event;
SUBCLASS OF ( Resource_event );
END_CLASS;
```

Class **Defined state relationship** *[urn:plcs:rdl:std#Defined_state_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Defined_state_relationship is a relationship that links a State_assertion to a State_assessment.

```
CLASS Defined_state_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Derived unit** *[urn:plcs:rdl:std#Derived_unit] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Derived_unit is a type of Unit that is defined by an expression of other units. EXAMPLE: Newtons per square metre is a derived unit.

```
CLASS Derived_unit;
SUBCLASS OF ( Unit );
END_CLASS;
```

Class **Descriptive document property** *[urn:plcs:rdl:std#Descriptive_document_property] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Descriptive_document_property is a type of String_representation_item that specifies a text based characteristics of a Document_definition or of a File.

```
CLASS Descriptive_document_property;
SUBCLASS OF ( String_representation_item );
END_CLASS;
```

Class **Detailed geometric model element** *[urn:plcs:rdl:std#Detailed_geometric_model_element] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Detailed_geometric_model_element is a type of Representation_item. It identifies a geometric construct. Only non abstract specializations of the Detailed_geometric_model_element entity data type can be instantiated.

```
CLASS Detailed_geometric_model_element;
SUBCLASS OF ( Representation_item );
SUPERCLASS OF ( Transformation_based_template_instance,
Mapping_based_template_instance, Cartesian_transformation_2d, Axis_placement,
Cartesian_transformation_3d, Cartesian_point, Direction, );
END_CLASS;
```

Class **Digital document definition** *[urn:plcs:rdl:std#Digital_document_definition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Digital_document_definition is a type of Document_definition. It identifies a collection of files that are archived on an optical computer disc, magnetic, electronic storage, or a combination thereof. A digital document definition may consist of one or many component digital files.

```
CLASS Digital_document_definition;
SUBCLASS OF ( Document_definition );
END_CLASS;
```

Class **Digital file** *[urn:plcs:rdl:std#Digital_file] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Digital_file is a type of File. A Digital_file contains computer interpretable data and is stored on an electronic device.

```
CLASS Digital_file;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Directed activity** *[urn:plcs:rdl:std#Directed_activity] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Directed_activity is a type of Activity. It identifies an actvity that is governed by a Work_order.

```
CLASS Directed_activity;
SUBCLASS OF ( Activity );
END_CLASS;
```

Class **Direction** *[urn:plcs:rdl:std#Direction] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Direction is a type of Detailed_geometric_model_element that defines a 2 or 3 dimensional vector. NOTE: A Direction is not located in a geometric space but is used in the definition of geometric entities like Axis_placement.

```
CLASS Direction;
SUBCLASS OF ( Detailed_geometric_model_element );
```

END_CLASS;

Class **Distribution by value** *[urn:plcs:rdl:std#Distribution_by_value] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Distribution_by_value is a type of Probability_distribution that explicitly lists pairs of random variable values and function values. NOTE: Distribution_by_value is used where there is no named distribution which can be used to identify the distribution, for example, when the distribution is derived from observation.

CLASS **Distribution_by_value**;
SUBCLASS OF (Probability_distribution);
END_CLASS;

Class **Document** *[urn:plcs:rdl:std#Document] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Document is a type of Product used to identify documentation data that is under configuration change management.

CLASS **Document**;
SUBCLASS OF (Product);
END_CLASS;

Class **Document assignment** *[urn:plcs:rdl:std#Document_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Document_assignment is a mechanism to associate a document with product data, where the assigned document provides information about the data with which it is associated.

CLASS **Document_assignment**;
SUBCLASS OF (PLCS-ARM-LF-THING);
SUPERCLASS OF (Partial_document_assignment,);
END_CLASS;

Class **Document definition** *[urn:plcs:rdl:std#Document_definition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Document_definition is a type of Product_view_definition that is a Document_version in a particular format. NOTE: A Document_version may have more than one representation. EXAMPLE: A version of a logical document, which contains a shape model, may be represented in the native formats of different CAD systems. Each Document_definition is a Digital_document_definition or a Physical_document_definition. NOTE: Aspects of the representation may not be known at the time the identification is established.

CLASS **Document_definition**;
SUBCLASS OF (Product_view_definition);
SUPERCLASS OF (Digital_document_definition, Physical_document_definition,);
END_CLASS;

Class **Document definition relationship** *[urn:plcs:rdl:std#Document_definition_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Document_definition_relationship is a relationship between two instances of Document_definition.

```
CLASS Document_definition_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Document location identification** *[urn:plcs:rdl:std#Document_location_identification] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Document_location_identification is a type of External_source_identification that identifies the location of the components of a Document_definition in an external storage system where they can be found. EXAMPLE: An HTML file that includes a picture may be represented as a Document_definition made of two components: * the HTML file; * the binary file that contains the picture. If these files were located within the same directory or relatively to the same directory, the External_source_identification.source_id attribute would convey the directory name. EXAMPLE: Examples of External_source_identification.source_type are: * 'URL' - for a web page; * 'FTP' - for an FTP address; * 'ISBN' - for physical documents.

```
CLASS Document_location_identification;
SUBCLASS OF ( External_source_identification );
END_CLASS;
```

Class **Document property representation** *[urn:plcs:rdl:std#Document_property_representation] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Document_property_representation is a type of Representation that conveys the parameters of a particular aspect of an Assigned_document_property. Where applicable, the following values shall be used for the inherited name attribute of the Document_property_representation : * 'document content': a category of parameters that precise characteristics of the data contained in a document definition or in a file; * 'document creation': a category of parameters that detail the context of creation of a document definition or file; * 'document format': a category of parameters that describe the format of a document definition or of a file; * 'document size': a category of parameters that detail the size of a document definition or of a file. The following characteristics may be items of a Document_property_representation with name 'document content': * A Descriptive_document_property with name 'detail level' specifying the level of detail that is provided; EXAMPLE: An example value for the 'detail level' property is 'rough 3d shape' - a 3D shape model without edge rounds and fillets. * A Descriptive_document_property with name 'geometry type' specifying the kind or kinds of geometry that is contained; EXAMPLE: examples of geometry type are: '3D wireframe model', '2D shape', 'surface model', 'closed volume', 'solid model', 'solid and surface model', '2D drawing'. * A Descriptive_document_property with name 'real world scale' specifying a text representation of the scale used. The following characteristics shall be an item of a Document_property_representation with name 'document creation': * A Descriptive_document_property with name 'creating system' specifying the name of the computer application or the machine that was used to generate the document definition or the file. The following characteristics may be items of a Document_property_representation with name 'document creation': * A Descriptive_document_property with name 'operating system' specifying the

name of the operating system that was used; * A Descriptive_document_property with name 'creating interface' specifying the name of the computer application used. EXAMPLE: An example value for the 'creating interface' property is 'Postscript Printer Driver' indicating that a Postscript Printer Driver was used to print a hardcopy. The following characteristics may be items of a Document_property_representation with name 'document format': * A Descriptive_document_property with name 'data format' specifying the name of the standard that defines the data structure used. Where applicable the following values shall be used: 'DXF', 'IGES', 'ISO 10303-203', 'ISO 10303-214', 'TIFF CCITT GR4','VDAFS','VOXEL'; * A Descriptive_document_property with name 'character code' specifying the character code used. Where applicable the following values shall be used: 'US ASCII 7bit', 'ISO 646', 'ISO 8859-1', 'EBCDIC', 'binary'; * A Descriptive_document_property with name 'size format standard' specifying the standard size used. EXAMPLE: 'A0' and 'A4' are examples of standard sizes of paper sheets. The following characteristics may be items of a Document_property_representation with name 'document size': * A Numerical_document_property with name 'page count' specifying the number of pages in a Physical_document_definition or a Hardcopy; * A Numerical_document_property with name 'file size' specifying the size of a digital file or of a Digital_document_definition.

```
CLASS Document_property_representation;
SUBCLASS OF ( Representation );
END_CLASS;
```

Class **Document version** *[urn:plcs:rdl:std#Document_version] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Document_version is a type of Product_version A Document_version identifies a particular version of a document.

```
CLASS Document_version;
SUBCLASS OF ( Product_version );
END_CLASS;
```

Class **Duration** *[urn:plcs:rdl:std#Duration] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Duration is a type of Value_with_unit that defines the magnitude of a time interval.

```
CLASS Duration;
SUBCLASS OF ( Value_with_unit );
END_CLASS;
```

Class **Effectivity** *[urn:plcs:rdl:std#Effectivity] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Effectivity is the identification of a domain of applicability. NOTE: Instances of Effectivity may be applied to any kind of product or activity data, using the constructs defined in the Effectivity application module.

```
CLASS Effectivity;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Lot_effectivity, Product_as_individual_effectivity,
Serial_effectivity, Time_interval_effectivity, Dated_effectivity, );
```

END_CLASS;

Class **Effectivity assignment** *[urn:plcs:rdl:std#Effectivity_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Effectivity_assignment is the association of an Effectivity with product or activity data.

```
CLASS Effectivity_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Effectivity relationship** *[urn:plcs:rdl:std#Effectivity_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Effectivity_relationship is an association between two instances of Effectivity. The meaning of the relationship is represented with the relation_type attribute.

```
CLASS Effectivity_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Electric current unit** *[urn:plcs:rdl:std#Electric_current_unit] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Electric_current_unit is a type of Unit with which movement of electrically charged particles is expressed.

```
CLASS Electric_current_unit;
SUBCLASS OF ( Unit );
END_CLASS;
```

Class **Element constraint** *[urn:plcs:rdl:std#Element_constraint] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Element_constraint is a type of Task_element_relationship that signifies a constraint between Task_element s. The constraint may only apply within the context of a Task_method or Task_element, specified as the context.

```
CLASS Element_constraint;
SUBCLASS OF ( Task_element_relationship );
END_CLASS;
```

Class **End task** *[urn:plcs:rdl:std#End_task] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An End_task is a type of Task_element. It signifies a point at which to end the task.

```
CLASS End_task;
SUBCLASS OF ( Task_element );
END_CLASS;
```

Class **Envelope** *[urn:plcs:rdl:std#Envelope] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Envelope is an historical record of the transmission of a Message. It is used to record the audit data for sending and acknowledging a Message. Because it is an historical record, each Envelope is only used once, and so is not versioned.

```
CLASS Envelope;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Envelope relationship** *[urn:plcs:rdl:std#Envelope_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Envelope_relationship is an association of one Envelope with another. NOTE: The most obvious use of the association is to link an Envelope to its acknowledgements.

```
CLASS Envelope_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Event** *[urn:plcs:rdl:std#Event] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Event is the fact of the existence of a state at some point in time. NOTE: The event may have occurred or may be not yet started. NOTE: The point in time where an event will start or started, may not be known or specified.

```
CLASS Event;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Relative_event, );
END_CLASS;
```

Class **Event assignment** *[urn:plcs:rdl:std#Event_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Event_assignment is assignment of an Event to product or activity data.

```
CLASS Event_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Event relationship** *[urn:plcs:rdl:std#Event_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Event_relationship is a relationship which exists between two Event s. EXAMPLE: An instance of this entity data type with Event_relationship.relation_type 'sequence' may be used to specify that an event shall end before another event starts.

```
CLASS Event_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Exit loop** *[urn:plcs:rdl:std#Exit_loop] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Exit_loop is a type of Structured_task_element. It signifies a point at which to exit from a loop.

```
CLASS Exit_loop;
SUBCLASS OF ( Task_element );
END_CLASS;
```

Class **Experience gained** *[urn:plcs:rdl:std#Experience_gained] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Experience_gained is a relationship between an Experience_instance and a person or organization.

```
CLASS Experience_gained;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Experience instance** *[urn:plcs:rdl:std#Experience_instance] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Experience_instance is a particular eposide of contact with and/or observation of facts or events which contributes to the accumulation of knowledge or skill. EXAMPLE: Changing the exhaust system on a car. EXAMPLE: 2 years work on same type of milling machine. EXAMPLE: 100 flying hours in a Tornado jet. EXAMPLE: 5 years as Workshop Manager. The nature, duration and worth of the Experience_instance can be described using assigned properties or by referring to activities or tasks.

```
CLASS Experience_instance;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Experience type** *[urn:plcs:rdl:std#Experience_type] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Experience_type is a category or class of experience. EXAMPLE: welding EXAMPLE: mechanical design EXAMPLE: flying

```
CLASS Experience_type;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **External class** *[urn:plcs:rdl:std#External_class] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An External_class is a type of Class that represents a reference to a class that is not included in the data exchange file and is defined in an external class library.

```
CLASS External_class;
SUBCLASS OF ( Class );
END_CLASS;
```

Class **External class library** *[urn:plcs:rdl:std#External_class_library] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An External_class_library is the identification of an external class library.

```
CLASS External_class_library;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **External geometric model** *[urn:plcs:rdl:std#External_geometric_model] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An External_geometric_model is a type of 3D Geometric_model whose content is provided in an external file. The set of items of the External_geometric_model contains only one element that shall be an Axis_placement. This Axis_placement specifies the reference location and orientation with respect to which placement of the External_geometric_model in other Geometric_model s shall be defined.

```
CLASS External_geometric_model;
SUBCLASS OF ( Geometric_model );
END_CLASS;
```

Class **External item identification** *[urn:plcs:rdl:std#External_item_identification] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An External_item_identification is a type of External_source_identification that provides the identifier of an item in the context of an external source where it can be found.

```
CLASS External_item_identification;
SUBCLASS OF ( External_source_identification );
SUPERCLASS OF ( File_location_identification, );
END_CLASS;
```

Class **External source identification** *[urn:plcs:rdl:std#External_source_identification] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An External_source_identification is the identification of the source where an item, or the components of an item, can be found.

```
CLASS External_source_identification;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Document_location_identification, External_item_identification,
);
END_CLASS;
```

Class File location identification *[urn:plcs:rdl:std#File_location_identification] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A File_location_identification is a type of External_item_identification that identifies the location of a File in an external storage system where it can be found. EXAMPLE: For a computer file identified by a filename and directory path, for example 'D:\project1\specification.txt', the External_item_identification.external_id attribute represents the filename, 'specification.txt' and the External_source_identification.source_id attribute represents the path name, 'D:\project1\'. EXAMPLE: Examples of External_source_identification.source_type are: * 'URL' - for a web page; * 'FTP' - for an FTP address; * 'ISBN' - for physical documents.

```
CLASS File_location_identification;
SUBCLASS OF ( External_item_identification );
END_CLASS;
```

Class File relationship *[urn:plcs:rdl:std#File_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A File_relationship is a relationship between two instances of File. EXAMPLE: A service manual may contain graphics for explanatory reasons. In this case, the File objects that contain the graphics are referenced as related from the File object that contains the body of the service manual with Document_definition_relationship.relation_type 'reference'.

```
CLASS File_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class Functional breakdown *[urn:plcs:rdl:std#Functional_breakdown] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Functional_breakdown is a type of Breakdown that identifies a partitioning of a product into a set of related functional elements so as to form explicit structural views that comprise the product elements. The composite - component view is represented by Functional_element_usage instances relating the functional elements in the breakdown which are represented by Functional_element s. EXAMPLE: A functional breakdown provides a decomposition of an aircraft in terms of high-level functional processes such as flight, taxiing and at rest all the way down to low-level processes such as detect onboard fuel level, move tail rudder and provide standard tow attachment point.

```
CLASS Functional_breakdown;
SUBCLASS OF ( Breakdown );
END_CLASS;
```

Class **Functional breakdown context** *[urn:plcs:rdl:std#Functional_breakdown_context] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Functional_breakdown_context is a type of Breakdown_context that is an association of a Functional_element to a Functional_breakdown of which the functional element is a member. EXAMPLE: A function 'provide load lifting capability' is member of the functional breakdown of a helicopter.

```
CLASS Functional_breakdown_context;
SUBCLASS OF ( Breakdown_context );
END_CLASS;
```

Class **Functional breakdown version** *[urn:plcs:rdl:std#Functional_breakdown_version] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Functional_breakdown_version is a type of Breakdown_version that identifies a version of a Functional_breakdown.

```
CLASS Functional_breakdown_version;
SUBCLASS OF ( Breakdown_version );
END_CLASS;
```

Class **Functional element** *[urn:plcs:rdl:std#Functional_element] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Functional_element is a type of Breakdown_element that identifies the elements in one or more Functional_breakdown objects.

```
CLASS Functional_element;
SUBCLASS OF ( Breakdown_element );
END_CLASS;
```

Class **Functional element definition** *[urn:plcs:rdl:std#Functional_element_definition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Functional_element_definition is a type of Breakdown_element_definition that identifies a view of a version (Functional_element_version) of a Functional_element.

```
CLASS Functional_element_definition;
SUBCLASS OF ( Breakdown_element_definition );
END_CLASS;
```

Class **Functional element usage** *[urn:plcs:rdl:std#Functional_element_usage] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Functional_element_usage is a type of Breakdown_element_usage that is an association of a functional element to another functional element that is a constituent. Functional_element.

CLASS **Functional_element_usage**;
SUBCLASS OF (Breakdown_element_usage);
END_CLASS;

Class **Functional element version** *[urn:plcs:rdl:std#Functional_element_version] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Functional_element_version is a type of Breakdown_element_version that identifies a version of a Functional_element.

CLASS **Functional_element_version**;
SUBCLASS OF (Breakdown_element_version);
END_CLASS;

Class **Geometric coordinate space** *[urn:plcs:rdl:std#Geometric_coordinate_space] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Geometric_coordinate_space is a type of Numerical_representation_context that defines a coordinate space where geometric elements can be defined. It is either two-dimensional or three-dimensional. There shall be at least two units specified for the Geometric_coordinate_space : one length unit and one plane angle unit. The length unit applies to each coordinate direction. EXAMPLE: The length unit millimetre and the angle unit radian are examples of units that may assigned to a Geometric_coordinate_space. NOTE: The origin for coordinate values is implicitly defined as being the cartesian point whose coordinates are all zero.

CLASS **Geometric_coordinate_space**;
SUBCLASS OF (Numerical_representation_context);
END_CLASS;

Class **Geometric model** *[urn:plcs:rdl:std#Geometric_model] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Geometric_model is a type of Representation dedicated to the description of geometric constructs. The Geometric_model is founded in a Geometric_coordinate_space. The items of a Geometric_model are instances of Detailed_geometric_model_element.

CLASS **Geometric_model**;
SUBCLASS OF (Representation);
SUPERCLASS OF (External_geometric_model,);
END_CLASS;

Class **Global location representation** *[urn:plcs:rdl:std#Global_location_representation] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Global_location_representation is a type of Location_representation specified using geographic means in the global system and values, which could be physical or political geographic values.

CLASS **Global_location_representation**;
SUBCLASS OF (Location_representation);

END_CLASS;

Class **Hardcopy** *[urn:plcs:rdl:std#Hardcopy] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Hardcopy is a type of File that represents a non-digital document. EXAMPLE: An actual stack of paper consisting of one or more sheets, on which some product data is written, printed or plotted.

```
CLASS Hardcopy;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Hierarchical interface connection** *[urn:plcs:rdl:std#Hierarchical_interface_connection] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Hierarchical_interface_connection is a type of Interface_connection that provides an interconnection between components at different levels in an assembly. Each connection point in the assembly is represented by a Interface_connector_occurrence. EXAMPLE: An appliance such as a television has a power lead and attached plug. The plug and power lead could be represented as an assembly of parts such as the plug pins and wires. Each connection point of the pins and wires in the assembly is represented by a Interface_connector_occurrence and an instance of the Hierarchical_interface_connection entity data type identifies the connection of the pins (the parts) to the plug (the assembly) in the assembly.

```
CLASS Hierarchical_interface_connection;
SUBCLASS OF ( Interface_connection );
END_CLASS;
```

Class **Hybrid breakdown** *[urn:plcs:rdl:std#Hybrid_breakdown] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Hybrid_breakdown is a type of Breakdown that identifies a non-specific partitioning of a product into a set of related elements so as to form explicit, parent-child views that comprise the elements. The parent-child view is represented by Hybrid_element_usage instances relating the elements in the hybrid breakdown which are represented by types of Breakdown_element s. EXAMPLE: A product breakdown in which a 'climate control' function has a decomposition that comprises a 'heating function' and a 'cooling function' and in which the 'heating function' has a decomposition that comprises a 'heating element' and a 'heat distribution system' would be an example of a 'hybrid breakdown'.

```
CLASS Hybrid_breakdown;
SUBCLASS OF ( Breakdown );
END_CLASS;
```

Class **Hybrid breakdown context** *[urn:plcs:rdl:std#Hybrid_breakdown_context] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Hybrid_breakdown_context is a type of Breakdown_context that is a relationship between a Breakdown_element and a Hybrid_breakdown of which the hybrid element is a member. EXAMPLE: A 'pipe element' might be a member of a 'heating function' breakdown in a hybrid decomposition.

```
CLASS Hybrid_breakdown_context;
SUBCLASS OF ( Breakdown_context );
END_CLASS;
```

Class Hybrid breakdown version *[urn:plcs:rdl:std#Hybrid_breakdown_version] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Hybrid_breakdown_version is a type of Breakdown_version that identifies a version of a Hybrid_breakdown.

```
CLASS Hybrid_breakdown_version;
SUBCLASS OF ( Breakdown_version );
END_CLASS;
```

Class Hybrid element usage *[urn:plcs:rdl:std#Hybrid_element_usage] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Hybrid_element_usage is a type of Breakdown_element_usage that identifies a relationship between a parent and child Breakdown_element where the parent and child elements may be different subtypes of Breakdown_element. EXAMPLE: In a hybrid breakdown, the 'fuel system' might include a 'tank element' and a 'fuel-injection element' as components.

```
CLASS Hybrid_element_usage;
SUBCLASS OF ( Breakdown_element_usage );
END_CLASS;
```

Class Identification assignment *[urn:plcs:rdl:std#Identification_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Identification_assignment is the assignment of an identifier to product or activity data.

```
CLASS Identification_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Alias_identification, );
END_CLASS;
```

Class In zone *[urn:plcs:rdl:std#In_zone] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An In_zone is a relationship between a Zone_element and an item that exists within the zone. EXAMPLE: A pump is in the starboard engine room of a ship.

```
CLASS In_zone;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Increasing resource event** *[urn:plcs:rdl:std#Increasing_resource_event] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Increasing_resource_event is a type of Resource_event that increases the balance of a managed resource. EXAMPLE: Purchasing new stock increases an inventory.

```
CLASS Increasing_resource_event;
SUBCLASS OF ( Resource_event );
END_CLASS;
```

Class **Independent property** *[urn:plcs:rdl:std#Independent_property] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Independent_property is a type of property. NOTE: It is independent of its application to characterize a product or an activity. EXAMPLE: The physical property kinematic viscosity, defined in ISO 31, is an example of Independent_property.

```
CLASS Independent_property;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Independent property relationship** *[urn:plcs:rdl:std#Independent_property_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Independent_property_relationship is a relationship between two instances of Independent_property. EXAMPLE: An Independent_property_relationship may be used to indicate that the values of a Independent_property can be derived from the another Independent_property.

```
CLASS Independent_property_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Independent property representation** *[urn:plcs:rdl:std#Independent_property_representation] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Independent_property_representation is the association of an Independent_property with a Representation.

```
CLASS Independent_property_representation;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Information right** *[urn:plcs:rdl:std#Information_right] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Information_right is a definition what may or may not be done with an item of information in the sense of legal rights and obligations. EXAMPLE: Copyright is an Information_right. EXAMPLE: For the purposes of developing a new system, details of government furnished equipment may be made available to a particular project team. The team may copy and use the information internally, but may not pass it on, either to a third part, or to another team, and must destroy the information at the end of the contract.

```
CLASS Information_right;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Information usage right** *[urn:plcs:rdl:std#Information_usage_right] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Information_usage_right is an application an Information_right to a particular usage context. NOTE: One view of the distinction between an Information_right and an Information_usage_right is that the Information_right represents a standard clause in a contract, whereas an Information_usage_right represents the fact that the clause is used in a particular contract. NOTE: The context for the usage can be defined through the contract which defines the right, the organization that grants the right, the person or organization which is granted the right, and any dates such as the starting or ending dates for the right. The meaning of each association is identified through the roles of the assignments, and these are defined through reference data. NOTE: The Approval of an Information_usage_right carries the meaning that the right is granted to all information items in the relevant context, as opposed to the approval of an Applied_information_usage_right which is limited to particular items. NOTE: Information_usage_right provides a mechanism for recording significant rights within a product database. The legal significance of the presence or absence of a right is outside the scope of this part of ISO 10303

```
CLASS Information_usage_right;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Information usage right relationship** *[urn:plcs:rdl:std#Information_usage_right_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Information_usage_right_relationship is the relationship of one Information_usage_right to another. EXAMPLE: Where one Information_usage_right supercedes another, then the original right is pointed to by the relating attribute, its replacement by the related attribute, and the relation_type attribute takes the value "supercedes".

```
CLASS Information_usage_right_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Interface connection** *[urn:plcs:rdl:std#Interface_connection] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Interface_connection is an interconnection between a connected pair of Interface_connector_occurrence s. Each Interface_connector_occurrence represents the place where a product used in an assembly can interact with other products in the assembly. EXAMPLE: An appliance

such as a television has a power lead and attached plug. The plug and power lead could be represented as an assembly of parts such as the plug pins and wires. Each connection point of the pins and wires in the assembly is represented by a Interface_connector_occurrence and an instance of the Interface_connection entity data type identifies the connection of the pins to the wires in the assembly.

```
CLASS Interface_connection;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Hierarchical_interface_connection, );
END_CLASS;
```

Class **Interface connector** *[urn:plcs:rdl:std#Interface_connector] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Interface_connector is a type of Product that identifies a part of a product with which one or more other products or the environment interacts. NOTE: This is sometimes referred to as a "port". EXAMPLE: A computer has a socket to which to connect a network cable. An instance of the Interface_connector entity data type identifies the role of the socket in the interface and is the subject of a specification that defines the necessary geometrical and electrical attributes to ensure a functioning interface between the computer and network hardware.

```
CLASS Interface_connector;
SUBCLASS OF ( Product );
END_CLASS;
```

Class **Interface connector as planned** *[urn:plcs:rdl:std#Interface_connector_as_planned] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Interface_connector_as_planned is a type of Interface_connector_version that identifies an individual that is the subject of a plan to realize an Interface_connector. EXAMPLE: Company Acme Limited is planning to produce an aircraft with serial number 1234 next month. This aircraft has connectors on each engine for fuel pipes. The company wishes to plan when each connector will be realized and then identify a date on which an inspector can test all the realized connectors.

```
CLASS Interface_connector_as_planned;
SUBCLASS OF ( Interface_connector_version );
END_CLASS;
```

Class **Interface connector as realized** *[urn:plcs:rdl:std#Interface_connector_as_realized] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Interface_connector_as_realized is a type of Interface_connector_version that identifies an individual that is a realized Interface_connector. EXAMPLE: Company WeFlySafest Corporation owns and operates an aircraft with serial number 1234. When landing at Heathrow airport, the pilot reports a loss of fuel pressure on engine number 4 with serial number A9876 and recommends that an inspector tests the realized connector on the engine.

```
CLASS Interface_connector_as_realized;
SUBCLASS OF ( Interface_connector_version );
END_CLASS;
```

Class **Interface connector definition** *[urn:plcs:rdl:std#Interface_connector_definition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Interface_connector_definition is a type of Product_view_definition that identifies a view of an Interface_connector. EXAMPLE: A reliability engineer assesses the likely failure modes of design version 3.8 for the connector between a brake unit and the hydraulic control system. The engineer generates a set of data that is a specific view of the connector. An instance of the Interface_connector_definition entity data type collects these data together.

```
CLASS Interface_connector_definition;
SUBCLASS OF ( Product_view_definition );
END_CLASS;
```

Class **Interface connector design** *[urn:plcs:rdl:std#Interface_connector_design] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Interface_connector_design is a type of Interface_connector_version that identifies a design version of an Interface_connector. EXAMPLE: BuildAWidget Incorporated creates design version 2.10 for an electrical supply connector.

```
CLASS Interface_connector_design;
SUBCLASS OF ( Interface_connector_version );
END_CLASS;
```

Class **Interface connector design to planned** *[urn:plcs:rdl:std#Interface_connector_design_to_planned] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Interface_connector_design_to_planned is a relationship between a design version of an Interface_connector and a planned individual that is to conform to the design. EXAMPLE: BuildAWidget Incorporated plans production of pump serial number 30301 with an electrical supply connector that is to conform to design version 2.10.

```
CLASS Interface_connector_design_to_planned;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Interface connector design to realized** *[urn:plcs:rdl:std#Interface_connector_design_to_realized] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Interface_connector_design_to_realized is a relationship between a design version of an Interface_connector and a realized individual that conforms to the design. EXAMPLE: BuildAWidget Incorporated builds pump serial number 30301 with an electrical supply connector that conforms to design version 2.11.

```
CLASS Interface_connector_design_to_realized;
SUBCLASS OF ( PLCS-ARM-LF-THING );
```

END_CLASS;

Class Interface connector occurrence *[urn:plcs:rdl:std#Interface_connector_occurrence] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Interface_connector_occurrence is an occurrence of a Interface_connector_definition.
The Interface_connector_occurrence represents the place where a product used in an assembly can interact with other products in the assembly. The interaction is represented by a Interface_connection.

CLASS **Interface_connector_occurrence;**
SUBCLASS OF (PLCS-ARM-LF-THING);
END_CLASS;

Class Interface connector planned to realized *[urn:plcs:rdl:std#Interface_connector_planned_to_realized] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Interface_connector_planned_to_realized is a relationship between a realized individual of an Interface_connector and a corresponding planned individual. EXAMPLE: BuildAWidget Incorporated builds pump serial number 30302 with an electrical supply connector that was previously planned.

CLASS **Interface_connector_planned_to_realized;**
SUBCLASS OF (PLCS-ARM-LF-THING);
END_CLASS;

Class Interface connector version *[urn:plcs:rdl:std#Interface_connector_version] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Interface_connector_version is a type of Product_version that identifies a version of an Interface_connector.

CLASS **Interface_connector_version;**
SUBCLASS OF (Product_version);
SUPERCLASS OF (Interface_connector_as_planned, Interface_connector_as_realized, Interface_connector_design,);
END_CLASS;

Class Interface definition connection *[urn:plcs:rdl:std#Interface_definition_connection] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Interface_definition_connection is an interconnection between a connected pair of Interface_connector_definition s or, if the point of interconnection is not specified, the interconnection between a pair of views (Product_view_definition s) on products EXAMPLE: A socket in the wall provides access to the domestic electricity supply. An appliance such as a television has a power lead and plug that fits into the socket. An instance of the Interface_definition_connection entity data type identifies this fitting of the plug into the socket.

CLASS **Interface_definition_connection;**
SUBCLASS OF (PLCS-ARM-LF-THING);

END_CLASS;

Class **Interface definition for** *[urn:plcs:rdl:std#Interface_definition_for] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Interface_definition_for is a relationship between an Interface_specification and an item that conforms to the specification. EXAMPLE: The infrared transmitter in a television remote control conforms to the specification that has the identifier 2345/XYZ/001. An instance of the Interface_definition_for entity data type is necessary to identify this relationship.

CLASS **Interface_definition_for**;
SUBCLASS OF (PLCS-ARM-LF-THING);
END_CLASS;

Class **Interface specification** *[urn:plcs:rdl:std#Interface_specification] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Interface_specification is a type of Product that provides a definition of necessary attributes for one or more items that participate in an interface. EXAMPLE: BSI develops a standard for connecting domestic electrical equipment to the electricity supply.

CLASS **Interface_specification**;
SUBCLASS OF (Product);
END_CLASS;

Class **Interface specification definition** *[urn:plcs:rdl:std#Interface_specification_definition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Interface_specification_definition is a type of Product_view_definition that provides a view of an Interface_specification. EXAMPLE: When developing a BSI standard for connecting domestic electrical equipment to the electricity supply, collected comments from experts form a new view on a version of the standard.

CLASS **Interface_specification_definition**;
SUBCLASS OF (Product_view_definition);
END_CLASS;

Class **Interface specification version** *[urn:plcs:rdl:std#Interface_specification_version] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Interface_specification_version is a type of Product_version that identifies a version of an Interface_specification. EXAMPLE: In 1999, BSI issues a new version of a standard for connecting domestic electrical equipment to the electricity supply.

CLASS **Interface_specification_version**;
SUBCLASS OF (Product_version);
END_CLASS;

Class **Item design association** *[urn:plcs:rdl:std#Item_design_association] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Item_design_association is the association of a Product_configuration with a Product_view_definition or a Product_version. It specifies the design that corresponds to the Product_configuration. If the design is a Product_view_definition, the Item_design_association represents the statement that, in the considered definition context, the product version, that is, the Product_view_definition is a valid way to implement the Product_configuration. NOTE: This association might not be valid in all definition contexts of the product version. If the design is a Product_version, the Item_design_association represents the statement that, in all definition contexts, the Product_version is a valid way to implement the Product_configuration. NOTE: The association might not be valid for other versions of the product.

```
CLASS Item_design_association;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Item shape** *[urn:plcs:rdl:std#Item_shape] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Item_shape is the shape of shapeable_item.

```
CLASS Item_shape;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Contextual_item_shape, );
END_CLASS;
```

Class **Item usage effectivity** *[urn:plcs:rdl:std#Item_usage_effectivity] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Item_usage_effectivity is an effectivity domain that constrains the use of a product with or within another product, in the context of a Product_configuration. The effectivity_domain attribute identifies a domain of effectivity. The item_usage_relationship attribute identifies a relationship which characterizes the use of the product with or within another product. EXAMPLE: This relationship may be an assembly-component relationship or a make-from relationship. The resolved_configuration attribute identifies an association between a Product_configuration and a product that implements it. This attribute establishes the context in which the item_usage_relationship is considered and constrained. When the effectivity domain is a range of serial numbers, the serial numbers considered are those of the Product_configuration. When the effectivity domain is defined using a production lot number, the production lot considered is one of the Product_configuration. When the effectivity domain is an interval of time, the interval of time considered is related to the production of the Product_configuration. NOTE: When no effectivity constraint is applied to a View_definition_usage, the validity or applicability status of this View_definition_usage is unknown.

```
CLASS Item_usage_effectivity;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Justification** *[urn:plcs:rdl:std#Justification] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Justification is the identification and description of the reasons for something. Justification entities may be associated with the data to which they apply. EXAMPLE: A justification may be provided for a product design. Similarly, a justification may be provided for why an activity is needed or was undertaken.

```
CLASS Justification;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Justification assignment** *[urn:plcs:rdl:std#Justification_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Justification_assignment is the association between a Justification and the item for which the Justification is provided. EXAMPLE: The item can be an activity or a product design.

```
CLASS Justification_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Justification relationship** *[urn:plcs:rdl:std#Justification_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Justification_relationship is an association between two Justification s.

```
CLASS Justification_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Justification support assignment** *[urn:plcs:rdl:std#Justification_support_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Justification_support_assignment is the association between a Justification and the item providing evidential support for the Justification. EXAMPLE: The support item can be the results from an analysis, a report, or professional judgment.

```
CLASS Justification_support_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Language** *[urn:plcs:rdl:std#Language] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Language is the identification of a language by its code as defined in ISO 639-2 and possibly its country code as specified in ISO 3166-1.

```
CLASS Language;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Language indication** *[urn:plcs:rdl:std#Language_indication] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Language_indication is the identification of the Language in which a text attribute of one or more instances has been specified. NOTE: The Language_indication entity data type identifies the primary language of a text attribute while the Attribute_translation_assignment entity data type enables to convey the translations of that attribute in various languages.

```
CLASS Language_indication;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Length unit** *[urn:plcs:rdl:std#Length_unit] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Length_unit is a type of Unit in which distances are expressed.

```
CLASS Length_unit;
SUBCLASS OF ( Unit );
END_CLASS;
```

Class **Local time** *[urn:plcs:rdl:std#Local_time] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Local_time is a point in time in a day, represented on a 24-hour clock by hour, minute and second. It is expressed in the local time zone and the offset from the Coordinate Universal Time shall be specified.

```
CLASS Local_time;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Location** *[urn:plcs:rdl:std#Location] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Location is a place or position where an activity or event can occur or a product or resource can exist.

```
CLASS Location;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Location assignment** *[urn:plcs:rdl:std#Location_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Location_assignment is a relationship between a product, event, or person and a location. There may be distinct assignment for each qualification. for example planned, scheduled or actual. Each assignment may have a start and end date or time. A location may have multiple assignments.

```
CLASS Location_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class Location relationship *[urn:plcs:rdl:std#Location_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Location_relationship is a relationship between two locations. EXAMPLE: Location B, which is in reference to location A or Location B (UK), which is a refinement of Location A (Europe).

```
CLASS Location_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class Location representation *[urn:plcs:rdl:std#Location_representation] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Location_representation is a means of representing a location.

```
CLASS Location_representation;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Global_location_representation,
Organization_based_location_representation,
Address_based_location_representation, Regional_grid_location_representation,
Product_based_location_identification, );
END_CLASS;
```

Class Looping element *[urn:plcs:rdl:std#Looping_element] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Looping_element is a type of Task_element. It invokes a specified number of repetitions of a further Task_element.

```
CLASS Looping_element;
SUBCLASS OF ( Structured_task_element );
SUPERCLASS OF ( Repeat_while, Repeat_until, Repeat_count, );
END_CLASS;
```

Class Lot effectivity *[urn:plcs:rdl:std#Lot_effectivity] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Lot_effectivity is a type of Effectivity for which the domain of applicability is defined as a given batch of items.

```
CLASS Lot_effectivity;
```

```
SUBCLASS OF ( Effectivity );
END_CLASS;
```

Class **Luminous intensity unit** *[urn:plcs:rdl:std#Luminous_intensity_unit] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Luminous_intensity_unit is a type of Unit in which the brightness of a body is expressed.

```
CLASS Luminous_intensity_unit;
SUBCLASS OF ( Unit );
END_CLASS;
```

Class **Make from relationship** *[urn:plcs:rdl:std#Make_from_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Make_from_relationship is a type of View_definition_usage established between two instances of Part_view_definition. It specifies that, in the context of the definition of the relating part version, the relating part version is considered as resulting from the manufacturing transformation of the related part version. NOTE: The related part version may identify a raw material or a semi-finished part. NOTE: The characteization of the process of transformation from the related part version to the relating part version it out of the scope of this application module.

```
CLASS Make_from_relationship;
SUBCLASS OF ( View_definition_usage );
END_CLASS;
```

Class **Managed resource** *[urn:plcs:rdl:std#Managed_resource] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Managed_resource is a representation of a resource that is provided with resource management capabilities. The role of a managed resource is determined by classification. EXAMPLE: A managed resource can be classified as "Stock line".

```
CLASS Managed_resource;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Managed resource relationship** *[urn:plcs:rdl:std#Managed_resource_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Managed_resource_relationship is an association between two managed resources. The meaning of the relationship is determined by classification. EXAMPLE: A managed resource relationship can be classified as "Alternative" or "Prefered".

```
CLASS Managed_resource_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Mapping based template instance** *[urn:plcs:rdl:std#Mapping_based_template_instance] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Mapping_based_template_instance is a type of Detailed_geometric_model_element. It is the replication of a Geometric_model. The replication transformation is defined by a Mapping_based_template_instance.source Axis_placement and a Mapping_based_template_instance.target that may be an Axis_placement or a cartesian_transformation. In the first case, the transformation shall be computed as the isometric transformation that maps: * the Axis_placement.origin of the Mapping_based_template_instance.source onto the Axis_placement.origin of the Mapping_based_template_instance.target, * the Axis_placement.x_axis of the Mapping_based_template_instance.source onto the Axis_placement.x_axis of the Mapping_based_template_instance.target, * the Axis_placement.y_axis of the Mapping_based_template_instance.source onto the Axis_placement.y_axis of the Mapping_based_template_instance.target. In the second case, the transformation shall be computed as the transformation that maps: * the Axis_placement.origin of the Mapping_based_template_instance.source onto the translation point of the Mapping_based_template_instance.target, * the Axis_placement.x_axis of the Mapping_based_template_instance.source onto the first element of the Mapping_based_template_instance.target of the Axis_placement_mapping.target, * the Axis_placement.y_axis of the Mapping_based_template_instance.source onto the second element of the Cartesian_transformation_2d.multiplication_matrix of the Axis_placement_mapping.target, * in case the transformation is a 3D transformation, the normalised vector product of the Axis_placement.x_axis and the Axis_placement.y_axis of the Mapping_based_template_instance.source onto the third element of the Cartesian_transformation_2d.multiplication_matrix of the Mapping_based_template_instance.target. NOTE: In the case where the Geometric_coordinate_space of the replicated model is not the same as the Geometric_coordinate_space in which the template_instance is founded, care shall be taken of any difference in the length units of both the instances of Geometric_coordinate_space, as unit conversion may be required. EXAMPLE: In a technical drawing of a mechanical part with several identical drilling holes, the hole geometry, is represented by a circle named 'annotated drilling hole'. This representation of the hole is replicated several times at different locations by corresponding template_instance objects. NOTE: The mapping-table defines two interpretations for this ARM entity. The first one uses the MIM entity mapped_item which uses the same principle to define the transformation as the Axis_placement_mapping and Axis_placement_transformation_mapping entities. The second one uses the MIM entity representation_relationship_with_transformation. Although this latter entity is not semantically equivalent to the ARM entity, the second interpretation is kept because of its use in some implementations.

```
CLASS Mapping_based_template_instance;
SUBCLASS OF ( Detailed_geometric_model_element );
END_CLASS;
```

Class **Market** *[urn:plcs:rdl:std#Market] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Market is the identification of a marketing segment for products.

```
CLASS Market;
```

SUBCLASS OF (PLCS-ARM-LF-THING);
END_CLASS;

Class **Mass unit** *[urn:plcs:rdl:std#Mass_unit] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Mass_unit is a type of Unit with which mass is expressed.

CLASS **Mass_unit**;
SUBCLASS OF (Unit);
END_CLASS;

Class **Measure item** *[urn:plcs:rdl:std#Measure_item] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Measure_item is a type of Representation_item that conveys the value of a quantity. Only specializations of this entity can be instantiated.

CLASS **Measure_item**;
SUBCLASS OF (Representation_item);
SUPERCLASS OF (Value_with_tolerances, Value_list, Measure_item_with_precision, Numerical_item_with_global_unit, Value_set, Value_range, Value_range_with_global_unit, Value_limit, Value_limit_with_global_unit, Numerical_item_with_unit,);
END_CLASS;

Class **Measure item with precision** *[urn:plcs:rdl:std#Measure_item_with_precision] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Measure_item_with_precision is a type of Measure_item that has a precision defined as the number of digits that are significant.

CLASS **Measure_item_with_precision**;
SUBCLASS OF (Measure_item);
END_CLASS;

Class **Message** *[urn:plcs:rdl:std#Message] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Message is a collection of information, brought together by an originator (the message definer) for some particular purpose, generally the fulfillment of a process. A Message is an historical record, intended to be sent using an Envelope and in consequence, is not versioned. NOTE: The same Message can be sent several times using different Envelope. Once it has been sent once, it cannot be further changed. However it should not remain unsent, since that is to confuse its functions and therefore its meaning with other entities such as one of the types of document.

CLASS **Message**;
SUBCLASS OF (PLCS-ARM-LF-THING);
END_CLASS;

Class **Message relationship** *[urn:plcs:rdl:std#Message_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Message_relationship is a link between two related messages. The meaning of the association depends on Message_relationship.role. EXAMPLE: If Message.id =2 replaces Message.id =1, then Message_relationship.related points to Message.id =2 and Message_relationship.relating points to Message.id =1 with Message_relationship.role ="replaces".

```
CLASS Message_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Next assembly usage** *[urn:plcs:rdl:std#Next_assembly_usage] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Next_assembly_usage is a type of Assembly_component_relationship. It establishes a relationship between a component and its immediate parent assembly in a product structure.

```
CLASS Next_assembly_usage;
SUBCLASS OF ( Assembly_component_relationship );
END_CLASS;
```

Class **Numerical document property** *[urn:plcs:rdl:std#Numerical_document_property] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Numerical_document_property is a type of Numerical_item_with_unit that specifies a numerical characteristic of a Document_definition or of a File. EXAMPLE: The size of a digital file expressed in Megabytes is an example of Numerical_document_property.

```
CLASS Numerical_document_property;
SUBCLASS OF ( Numerical_item_with_unit );
END_CLASS;
```

Class **Numerical item with global unit** *[urn:plcs:rdl:std#Numerical_item_with_global_unit] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Numerical_item_with_global_unit is a type of Measure_item where the value is expressed with respect to a unit provided as a global information associated with the representation context.

```
CLASS Numerical_item_with_global_unit;
SUBCLASS OF ( Measure_item );
SUPERCLASS OF ( Probability_function_value, Probability_distribution_parameter,
Probability_derivation_parameter, Probability_numeric_value, Random_variable, );
END_CLASS;
```

Class **Numerical item with unit** *[urn:plcs:rdl:std#Numerical_item_with_unit] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Numerical_item_with_unit is a type of Measure_item that is also a Value_with_unit. The quantity value is therefore provided with its own unit. NOTE: A unit that is specified by Numerical_item_with_unit supersedes a unit associated with the Numerical_representation_context.

```
CLASS Numerical_item_with_unit;
SUBCLASS OF ( Measure_item Value_with_unit );
SUPERCLASS OF ( Numerical_document_property, );
END_CLASS;
```

Class **Numerical representation context** *[urn:plcs:rdl:std#Numerical_representation_context] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Numerical_representation_context is a type of Representation_context in which units and uncertainties may be defined. These units and uncertainties apply to the instances of Representation_item that are used in that context.

```
CLASS Numerical_representation_context;
SUBCLASS OF ( Representation_context );
SUPERCLASS OF ( Geometric_coordinate_space, );
END_CLASS;
```

Class **Observation** *[urn:plcs:rdl:std#Observation] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Observation is an historical record of something that has occured during the life of a product or its support environment. Its use is restricted to observations not directly represented in the data model, and should not be used where some other reporting data structure is defined.

```
CLASS Observation;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Observation consequence** *[urn:plcs:rdl:std#Observation_consequence] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Observation_consequence is an association of an observation to the consequences that follow from it, where those consequences are in the form of a Work_request. NOTE: One of the uses of Observation_consequence is to close one of the feedback loops from the use of a product to requests for its enhancements.

```
CLASS Observation_consequence;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Observation item** *[urn:plcs:rdl:std#Observation_item] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Observation_item is a reference to any item that an observation can be made about, or any item that can be used as part of the observation. NOTE: An observation can refer to any entity represented in the schema it is used in, from products to property values, or actions to task steps. The mechanism used gives no interpretation of how the Observation_item.item_identifier is interpreted or how the data is accessed. For example, the identifier could be the number of the entity in a file conforming to ISO-10303-21.

```
CLASS Observation_item;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Observation relationship** *[urn:plcs:rdl:std#Observation_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Observation_relationship is a relationship between two observations. The nature of this relationship is identified by the role. Where there is a structural relationship between Observation, the semantics of the structure are identified by the classification of the Observation_relationship against reference data. EXAMPLE: The Observation of a persistant fault is composed of a series of Observation of occurances of the same fault. That is, Observation_relationship.related points to the composite Observation, while Observation_relationship.relating points to one actual Observation of the occurance.
The Observation_relationship.role of the relationship is "observed instance", while it is classified as "is composed of". In this example, the component Observation will apply to Product_as_realized and the consequence will be to rectify the individual faults, while the composite Observation will apply to a Product_version and the consequence will be a design change.

```
CLASS Observation_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Or state cause effect definition** *[urn:plcs:rdl:std#Or_state_cause_effect_definition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Or_state_cause_effect_definition is a type of State_cause_effect_definition. It relates one or more State_definition entities that are causes to a State_definition that is the effect. At least one cause must exist prior to the effect.

```
CLASS Or_state_cause_effect_definition;
SUBCLASS OF ( State_cause_effect_definition );
END_CLASS;
```

Class **Organization** *[urn:plcs:rdl:std#Organization] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Organization is an administrative structure in which persons are active.

```
CLASS Organization;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Organization based location representation**

[urn:plcs:rdl:std#Organization_based_location_representation] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Organization_based_location_representation is a type of Location_representation that specifies a location in the context of an organization. EXAMPLE: The location "Room 99" in "The Administration Building" of a particular university might be represented using one instance of Organization_based_location_representation with two instances of Organizational_location_identification and one instance of Organization.

```
CLASS Organization_based_location_representation;
SUBCLASS OF ( Location_representation );
END_CLASS;
```

Class **Organization or person in organization assignment**

[urn:plcs:rdl:std#Organization_or_person_in_organization_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Organization_or_person_in_organization_assignment is an association of an organization or a person in an organization with activity or product data.

```
CLASS Organization_or_person_in_organization_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Organization relationship** *[urn:plcs:rdl:std#Organization_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Organization_relationship is a relationship between two instances of Organization. EXAMPLE: A team belongs to a department which itself belongs to a company. Such an organizational structure can be described up using instances of Organization_relationship.

```
CLASS Organization_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Organization type** *[urn:plcs:rdl:std#Organization_type] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Organization_type is a recognized kind of Organization. EXAMPLE: legal entity EXAMPLE: change control board EXAMPLE: sales organization EXAMPLE: manufacturing organization EXAMPLE: department

```
CLASS Organization_type;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Organizational location identification**

[urn:plcs:rdl:std#Organizational_location_identification] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Organizational_location_identification is location that is defined by an identification which is specific in an organization context.

```
CLASS Organizational_location_identification;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Parameterized distribution** *[urn:plcs:rdl:std#Parameterized_distribution] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Parameterized_distribution is a type of Probability_distribution that is used to link a named probability distribution to the parameters that define it. NOTE: The Parameterized_distribution.parameterization_name of this entity is used to discriminate between alternative parameterizations of the same distribution. NOTE: To calculate a value using a Parameterized_distribution it is necessary to know the general distribution function, the value of the distribution function parameters, and the specific random variable value for which the probability evaluation is required. The Probability Distribution module does not define the formula for the distribution function, and it is assumed that this is defined externally, either through reference information or via a "formula" module. This entity provides the distribution function parameters. The value of the random variable for which the probability is calculated is provided by the Probability_derivation_parameter in the Probability module (see note 2).

```
CLASS Parameterized_distribution;
SUBCLASS OF ( Probability_distribution );
END_CLASS;
```

Class **Part** *[urn:plcs:rdl:std#Part] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Part is a type of Product that collects the definitional information of the versions of either a part or of a non-countable material. NOTE: A Part does not represent an actual physical object that is or was existing in the real world. NOTE: A complex instance of the Part entity and of the Document entity may be created in order to represent a document that is a component of a manufactured product, for example a user manual of a car.

```
CLASS Part;
SUBCLASS OF ( Product );
END_CLASS;
```

Class **Part version** *[urn:plcs:rdl:std#Part_version] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Part_version is a type of Product_version that identifies a version of a part. A Part_version serves as the collector of the data characterizing a realizable object in various application contexts. NOTE:

A Part_version is expected to be functionally and physically interchangeable with the other versions of the same Part.

```
CLASS Part_version;
SUBCLASS OF ( Product_version );
END_CLASS;
```

Class **Part view definition** *[urn:plcs:rdl:std#Part_view_definition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Part_view_definition is a type of Product_view_definition that defines a characterization view of a version of a Part.

```
CLASS Part_view_definition;
SUBCLASS OF ( Product_view_definition );
END_CLASS;
```

Class **Partial document assignment** *[urn:plcs:rdl:std#Partial_document_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Partial_document_assignment is a type of Document_assignment.
A Partial_document_assignment identifies a specific portion of the contents of a document. It assigns this portion to the product data for which it is relevant.

```
CLASS Partial_document_assignment;
SUBCLASS OF ( Document_assignment );
END_CLASS;
```

Class **Person** *[urn:plcs:rdl:std#Person] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Person is an individual human being.

```
CLASS Person;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Person in organization** *[urn:plcs:rdl:std#Person_in_organization] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Person_in_organization is the identification of a Person and of his role in an Organization

```
CLASS Person_in_organization;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Person or organization or person in organization in position** *[urn:plcs:rdl:std#Person_or_organization_or_person_in_organization_in_position] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Person_or_organization_or_person_in_organization_in_position is a Person, Organization, or Person_in_organization that holds a Position. A person may hold more than one position within an organization or organizations. EXAMPLE: A person can hold two positions in an organization: Production Manager and Safety Officer. A position in an organization can be held by more than one person or organization. EXAMPLE: Two people can hold the same position in a job-share scheme.

```
CLASS Person_or_organization_or_person_in_organization_in_position;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Person or organization or person in organization in position relationship** *[urn:plcs:rdl:std#Person_or_organization_or_person_in_organization_in_position_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Person_or_organization_or_person_in_organization_in_position_relationship is a relationship between one Organization, Person, or Person_in_organization and another. Examples of the relationship are EXAMPLE: successor EXAMPLE: job-share NOTE: The meaning of the relationship is determined by classification

```
CLASS Person_or_organization_or_person_in_organization_in_position_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Physical breakdown** *[urn:plcs:rdl:std#Physical_breakdown] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Physical_breakdown is a type of Breakdown that identifies a partitioning of a product into a set of related physical elements so as to form explicit, parent-child views that comprise the product elements. The parent-child view is represented by Physical_element_usage instances relating the physical elements in the breakdown which are represented by Physical_element s. EXAMPLE: A physical breakdown might provide a decomposition of an automobile in terms such as body, roof, bonnet, bumpers and this breakdown might be differerent from, and orthogonal to, a parts decomposition.

```
CLASS Physical_breakdown;
SUBCLASS OF ( Breakdown );
END_CLASS;
```

Class **Physical breakdown context** *[urn:plcs:rdl:std#Physical_breakdown_context] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Physical_breakdown_context is a type of Breakdown_context that is a membership relationship between a Physical_element and a Physical_breakdown of which the physical element is a member. EXAMPLE: A wheel is a member of the physical breakdown of an automobile.

```
CLASS Physical_breakdown_context;
SUBCLASS OF ( Breakdown_context );
END_CLASS;
```

Class **Physical breakdown version** *[urn:plcs:rdl:std#Physical_breakdown_version] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Physical_breakdown_version is a type of Breakdown_version that identifies a version of a Physical_breakdown. EXAMPLE: An engineer modifies the current physical breakdown for an aircraft on the basis of a tail re-design.

```
CLASS Physical_breakdown_version;
SUBCLASS OF ( Breakdown_version );
END_CLASS;
```

Class **Physical document definition** *[urn:plcs:rdl:std#Physical_document_definition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Physical_document_definition is a type of Document_definition. A physical document definition may consist of one or many component hardcopy files. EXAMPLE: Paper plots of technical drawings, micro fiche, or paper documents such as calculations or test reports are examples of Physical_document_definition.

```
CLASS Physical_document_definition;
SUBCLASS OF ( Document_definition );
END_CLASS;
```

Class **Physical element** *[urn:plcs:rdl:std#Physical_element] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Physical_element is a type of Breakdown_element that identifies the elements in one or more Physical_breakdown objects.

```
CLASS Physical_element;
SUBCLASS OF ( Breakdown_element );
END_CLASS;
```

Class **Physical element definition** *[urn:plcs:rdl:std#Physical_element_definition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Physical_element_definition is a type of Breakdown_element_definition that identifies a view of a version (Physical_element_version) of a Physical_element.

```
CLASS Physical_element_definition;
SUBCLASS OF ( Breakdown_element_definition );
END_CLASS;
```

Class **Physical element usage** *[urn:plcs:rdl:std#Physical_element_usage] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Physical_element_usage is a type of Breakdown_element_usage that is a relationship between a parent and child Physical_element. EXAMPLE: In a physical breakdown, the aircraft (parent) might include (as children) a fuselage, wings, tail and undercarriage.

```
CLASS Physical_element_usage;
SUBCLASS OF ( Breakdown_element_usage );
END_CLASS;
```

Class **Physical element version** *[urn:plcs:rdl:std#Physical_element_version] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Physical_element_version is a type of Breakdown_element_version that identifies a version of a Physical_element. EXAMPLE: An engineer changes the details describing the undercarriage that is an element in a physical breakdown of an aircraft.

```
CLASS Physical_element_version;
SUBCLASS OF ( Breakdown_element_version );
END_CLASS;
```

Class **Plane angle unit** *[urn:plcs:rdl:std#Plane_angle_unit] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Plane_angle_unit is a type of Unit by which a angles in planes are expressed.

```
CLASS Plane_angle_unit;
SUBCLASS OF ( Unit );
END_CLASS;
```

Class **PLCS-ARM-LF-THING** *[urn:plcs:rdl:std#PLCS-ARM-LF-THING] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

```
CLASS PLCS-ARM-LF-THING;

SUPERCLASS OF ( Project_assignment, Approval_relationship, Organization,
Requirement_source, Representation, Condition, Message_relationship,
Interface_connector_design_to_realized, Observation, State_definition,
Information_usage_right, Condition_evaluation_assignment,
Interface_connector_planned_to_realized, Product_view_definition,
Applied_activity_assignment, Approval_status, Justification_support_assignment,
Position_type_assignment, Information_usage_right_relationship,
Resource_property_representation, Effectivity_relationship,
Observation_relationship, Related_condition_parameter,
Product_planned_to_realized, Unit, Position_position_type_assignment,
Attribute_translation_assignment, Product_design_to_individual,
Alternate_part_relationship, Applied_state_definition_assignment,
Work_request_status, Contract_assignment, Product, Effectivity,
Breakdown_context, Effectivity_assignment, Independent_property, Location,
Defined_state_relationship, Condition_relationship, Location_representation,
Condition_parameter, Activity_method_realization, Activity_method,
Item_design_association, Independent_property_relationship,
```

Representation context, Attachment slot design to planned, Requirement assignment, Security classification, Interface definition for, Activity method relationship, Digital file, State relationship, Event relationship, Experience gained, Representation relationship, Required resource relationship, Interface connector occurrence, Work output assignment, Work output, Type of person assignment, State definition relationship, Activity relationship, Certification, Task element state relationship, Approving person organization, Position assignment, Product group membership, Resource item assignment, Envelope relationship, Item usage effectivity, Local time, Contract, Task method state relationship, Product design version to individual, Required resource assignment, Condition evaluation, Activity method realization relationship, Applied state assignment, Selected item assignment, Message, Representation item, Position relationship, Content item, Project relationship, Product concept, Address assignment, Attachment slot planned to realized, Resource item relationship, State, Breakdown of, Interface connector design to planned, Location relationship, Activity status, Activity property representation, Time interval relationship, Affected items assignment, Certification assignment, Task objective, Product relationship, Activity property, Project, Value with unit, Breakdown element realization, Organization relationship, Type of person definition required attributes relationship, Experience instance, Work order, Attachment slot on product, Interface connection, View definition relationship, Date or date time assignment, Person or organization or person in organization in position, Calendar date, Type of person, Location assignment, Product group, Resource item, Position, Managed resource, Required resource, Managed resource relationship, Applied information usage right, Qualification assignment, Type of person definition, Product version relationship, Security classification assignment, In zone, Language indication, Market, Product version, Information right, Observation consequence, Regional coordinate, Resource as realized, Position group, Date time, Work request, Approval assignment, Product category, Event assignment, Independent property representation, Person or organization or person in organization in position relationship, Task objective state relationship, Language, Property representation, Resource as realized assignment, Justification relationship, Assembly relationship substitution, File relationship, Interface definition connection, Justification assignment, Person, Class, State assertion, Qualification type, Product group relationship, Approval, View definition context, Qualification type relationship, Work output relationship, Resource property, Resource event correspondence relationship, External class library, Observation item, Experience type, External source identification, Condition assignment, Document assignment, Attachment slot design to realized, Item shape, Resource event relationship, Identification assignment, Document definition relationship, Hardcopy, Justification, Type of person definition relationship, Event, Organizational location identification, Address, Position group relationship, Position type, Person in organization, Position group assignment, Envelope, Characterizable object, State role, Organization type, Organization or person in organization assignment, Applied activity method assignment, Product configuration, Activity method assignment, Assigned property, Condition evaluation parameter,

State_assessment, Resource_as_realized_relationship, Resource_event, Activity,
);
END_CLASS;

Class **Position** *[urn:plcs:rdl:std#Position] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Position is a function or job performed by a person. It defines responsibilities and activities. A position that is not fulfilled by a person is a vacancy. EXAMPLE: Company Director EXAMPLE: Service Engineer

```
CLASS Position;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Position assignment** *[urn:plcs:rdl:std#Position_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Position_assignment is an association of a Position with activity or product data.

```
CLASS Position_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Position group** *[urn:plcs:rdl:std#Position_group] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Position_group is a group for collecting people in their positions to cooperate in an assignment. EXAMPLE: People representing utility companies who cooperate with each other to coordinate road works.

```
CLASS Position_group;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Position group assignment** *[urn:plcs:rdl:std#Position_group_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Position_group_assignment is an association of a Position_group with activity or product data.

```
CLASS Position_group_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Position group relationship** *[urn:plcs:rdl:std#Position_group_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Position_group_relationship is a relationship that specifies the participation of a position in a group.
CLASS **Position_group_relationship;**
SUBCLASS OF (PLCS-ARM-LF-THING);
END_CLASS;

Class **Position position type assignment** *[urn:plcs:rdl:std#Position_position_type_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Position_position_type_assignment is an assignment of Position_type to a Position. It allows many positions to be described by many position types.
CLASS **Position_position_type_assignment;**
SUBCLASS OF (PLCS-ARM-LF-THING);
END_CLASS;

Class **Position relationship** *[urn:plcs:rdl:std#Position_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Position_relationship is a relationship between Position s. Examples of relationships between Position s are EXAMPLE: Superior / Subordinate EXAMPLE: Superceded by NOTE: The meaning of the relationship is determined by classification.
CLASS **Position_relationship;**
SUBCLASS OF (PLCS-ARM-LF-THING);
END_CLASS;

Class **Position type** *[urn:plcs:rdl:std#Position_type] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Position_type is a recognized kind of position. EXAMPLE: Chief Executive Officer EXAMPLE: Manager EXAMPLE: Service engineer EXAMPLE: Programmer EXAMPLE: Aircraft engineering technician propulsion
CLASS **Position_type;**
SUBCLASS OF (PLCS-ARM-LF-THING);
END_CLASS;

Class **Position type assignment** *[urn:plcs:rdl:std#Position_type_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Position_type_assignment is the association of a Position_type with activity or product data.
CLASS **Position_type_assignment;**
SUBCLASS OF (PLCS-ARM-LF-THING);
END_CLASS;

Class **Probability** *[urn:plcs:rdl:std#Probability] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Probability is a type of Representation that is a probability value (see definition 3.5.1 above).

```
CLASS Probability;
SUBCLASS OF ( Representation );
SUPERCLASS OF ( Probability_numeric, Probability_by_name, );
END_CLASS;
```

Class **Probability by name** *[urn:plcs:rdl:std#Probability_by_name] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Probability_by_name is a type of Probability whose value belongs to a one of a set of named classes, rather than by assigning a specific numerical value, which may not be available. EXAMPLE: A safety assessment methodology classes the probability an accident as "very unlikely", "unlikely", "significantly likely" and "almost certain". Any process that has a "very likely" or "almost certain" chance of causing serious injury is shut down.

```
CLASS Probability_by_name;
SUBCLASS OF ( Probability );
END_CLASS;
```

Class **Probability derivation parameter** *[urn:plcs:rdl:std#Probability_derivation_parameter] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Probability_derivation_parameter is a type of Numerical_item_with_global_unit that is used by a Probability_derived in a particular role in order to calculate the particular probability. EXAMPLE: In a coin tossing trial, the probability calculated is that of getting more than 6 heads in ten tosses. The parameter with the role "minimum number of heads" will have the value "6" NOTE: The role name is given by the 'name' attribute inherited from Representation_item, and the set of such names and their interpretation is defined through reference data. NOTE: The value attribute, which holds the parameter value, is inherited from the supertype.

```
CLASS Probability_derivation_parameter;
SUBCLASS OF ( Numerical_item_with_global_unit );
END_CLASS;
```

Class **Probability derived** *[urn:plcs:rdl:std#Probability_derived] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Probability_derived is a type of Probability_numeric that associates a particular value of Probability_numeric with the source from which the value derived together with any parameters used to get the particular value. NOTE: Where the probability derived from a probability distribution, the parameters of Probability_derived are those needed to get a single value from the distibution, and not those which characterise the distribution. For example, in the case of coin tossing, the distribution is a Binomial distribution, with parameters of 'probability for a single toss', and 'number of tosses', whereas the parameter for the Probability_derived will be the 'number of heads obtained'.

```
CLASS Probability_derived;
```

SUBCLASS OF (Probability_numeric);
END_CLASS;

Class **Probability distribution** *[urn:plcs:rdl:std#Probability_distribution] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Probability_distribution is a type of Probability_generator that is a probability distribution. NOTE: For a full understanding of probability distribution and the terms used, a textbook on probability theory should be consulted. NOTE: This entity describes a particular probability distribution, rather than the general type of distribution. For example, in coin tossing experiment, the number of heads that may occur is given by a binomial distribution - that is, a type of distribution, and outside the scope of this module. This module provides the description of the distribution of a particular experiment, say, 10 tosses of a particular coin. The actual probability of an outcome, say 6 heads in 10 tosses, is recorded using the probability module. NOTE: The attributes 'name', 'id' and 'description' are inherited from the supertype Representation. The name provides a cue to the particular source of the distribution, such as "A fair coin tossed 10 times", rather than the type of distribution (in this case Binomial) which is given by the Probability_distribution.distribution_name attribute.

CLASS **Probability_distribution**;
SUBCLASS OF (Probability_generator);
SUPERCLASS OF (Parameterized_distribution, Distribution_by_value,);
END_CLASS;

Class **Probability distribution parameter** *[urn:plcs:rdl:std#Probability_distribution_parameter] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Probability_distribution_parameter is a type of Numerical_item_with_global_unit that is one of the set of values that characterises a probability distribution. EXAMPLE: The Normal (or Gaussian) distribution has, in the standard parameterisation, two parameters: the mean and the variance NOTE: For many common distributions, the mean and the variance are sufficient to characterize a distribution, and the parameter list may be empty. NOTE: Probability_distribution_parameter inherits the 'name' attribute from its supertype, and this is used to identify the name of the parameter within the particular parameterization. The value attribute is also inherited.

CLASS **Probability_distribution_parameter**;
SUBCLASS OF (Numerical_item_with_global_unit);
END_CLASS;

Class **Probability function value** *[urn:plcs:rdl:std#Probability_function_value] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Probability_function_value is a type of Numerical_item_with_global_unit that is the value of the probability function at the given random variable value. NOTE: The value is an inherited attribute. It is not in general a probability value. In some functions for continuous distributions, the probability that the random variable lies between two values is the integral of the function of that range.

CLASS **Probability_function_value**;
SUBCLASS OF (Numerical_item_with_global_unit);
END_CLASS;

Class **Probability generator** *[urn:plcs:rdl:std#Probability_generator] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Probability_generator is a type of Representation. It is a source from the Probability_derived is derived. The Probability_derivation_parameter s are applied to the Probability_generator to get the particular derived value. NOTE: A Probability_generator will generally be either a Probability_distribution or a function of some statistics. EXAMPLE: A probability of "0.67" is derived from a Normal (or Gaussian) distribution using the parameter "plus or minus '1.0' standard deviations from the mean"

```
CLASS Probability_generator;
SUBCLASS OF ( Representation );
SUPERCLASS OF ( Probability_distribution, );
END_CLASS;
```

Class **Probability named value** *[urn:plcs:rdl:std#Probability_named_value] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Probability_named_value is a type of Representation_item that is used to hold the name of the probability value. NOTE: The value attribute is the description inherited from the supertype. In general, this value will be one of an enumeration of possible values defined through reference data.

```
CLASS Probability_named_value;
SUBCLASS OF ( Representation_item );
END_CLASS;
```

Class **Probability numeric** *[urn:plcs:rdl:std#Probability_numeric] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Probability_numeric is a type of Probability that is expressed as a numeric value between 0 and 1.

```
CLASS Probability_numeric;
SUBCLASS OF ( Probability );
SUPERCLASS OF ( Probability_derived, );
END_CLASS;
```

Class **Probability numeric value** *[urn:plcs:rdl:std#Probability_numeric_value] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Probability_numeric_value is a type of Numerical_item_with_global_unit providing a numeric representation of a probability. NOTE: The value attribute is inherited from the supertype.

```
CLASS Probability_numeric_value;
SUBCLASS OF ( Numerical_item_with_global_unit );
END_CLASS;
```

Class **Product** *[urn:plcs:rdl:std#Product] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Product is the identification of a product or of a type of product. It is a collector of data common to all revisions of the Product. NOTE: Products that this entity data type can represent, include: * products existing in the real world; * products that may come into existence as a consequence of some realization process. This includes parts and documents; * products that are functions. In the interpreted models, these various meanings are represented within instances of the entity product_related_product_category, with prescribed values of the category name. For example, the category name 'document' shall be used when characterizing the fact that a product is actually a document. EXAMPLE: The SS Titanic is a product that could be represented by the entity data type Product. EXAMPLE: Lifeboat is a class of products that could be represented by the entity data type Product. Each lifeboat on the SS Titanic is a member of this class. NOTE: A product is identified by an organization or a person in an organization. The definition of the domain of uniqueness and the mechanism for guaranteeing the uniqueness of product id are outside the scope of this application module. NOTE: A product may have zero or more versions. A version of a product is represented with an instance of the entity Product_version or of one of its specializations.

```
CLASS Product;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Breakdown, Attachment_slot, Breakdown_element, Requirement,
Interface_specification, Product_as_individual, Part, Document,
Interface_connector, );
END_CLASS;
```

Class **Product as individual** *[urn:plcs:rdl:std#Product_as_individual] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Product_as_individual is a type of Product that identifies an individual artefact that has been made or is planned to be made. It is a collector of data common to all revisions of the Product_as_individual. NOTE: Revisions (Product_as_individual_version) of Product_as_individual are used to represent whether an individual artefact has yet to be made or has been made.

The Product_as_individual_version subtype Product_as_planned is used to represent the revision of an individual artefact that has yet to be made and the subtype Product_as_realized is used to represent the revision of an individual artefact that has been made. NOTE: Where physical products are being represented, the Product_as_individual represents the physical or planned physical realization of a product. NOTE: It is likely, but not essential, that the artefact, was or will be made from a product design. The product design will be represented by a Product which will be related to the Product_as_individual by Product_design_to_individual. NOTE: Many physical products may be produced from a given design. A single Product may lead to many Product_as_individual s. EXAMPLE: The design of a personal computer is represented by a Product. EXAMPLE: The personal computer with a serial number on a persons desk is represented by a Product_as_individual and an associated revision represented by Product_as_realized. EXAMPLE: The personal computer that has been ordered, allocated a serial number for manufacturing planning, but not yet manufactured, is represented by a Product_as_individual and an associated revision represented by Product_as_planned. EXAMPLE: HMS Daring is the first of a new class of ships known as Type 45 Destroyers. It is due to enter service in two years time. * The Type 45 ship design that applies to HMS Daring is represented by a Product, * The ship HMS Daring is represented by a Product_as_individual. This identifies the planned ship, and when built, the actual ship; * The planned revision of the future ship HMS Daring is represented by a Product_as_planned which is related back to the Product_as_individual, * When built, the first revision the actual ship, HMS Daring will be represented by a Product_as_realized which is related back to the Product_as_individual.

```
CLASS Product_as_individual;
SUBCLASS OF ( Product );
END_CLASS;
```

Class **Product as individual effectivity** *[urn:plcs:rdl:std#Product_as_individual_effectivity] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Product_as_individual_effectivity is a type of Effectivity for which the domain of applicability is defined as a set of instances of Product_as_individual.

```
CLASS Product_as_individual_effectivity;
SUBCLASS OF ( Effectivity );
END_CLASS;
```

Class **Product as individual version** *[urn:plcs:rdl:std#Product_as_individual_version] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Product_as_individual_version is a type of Product_version. It is a revision of a Product_as_individual and acts as a collector of the definitions of this revision. EXAMPLE: The car on my drive is represented by a Product_as_individual. The current configuration status of the car can be represented by a Product_as_realized related to the Product_as_individual. If a safety modification is made to the car resulting in a new configuration status of the car, then this may be represented by a new Product_as_realized. EXAMPLE: HMS Daring is the first of a new class of ships known as Type 45 Destroyers. It is due to enter service in two years time. * The version of the generic Type 45 ship design that applies to HMS Daring is represented by a Product_version, * The ship HMS Daring is represented by a Product_as_individual. This identifies the planned ship, and when built, the actual ship; * The planned revision of the future ship HMS Daring is represented by a Product_as_planned related back to the Product_as_individual, * When built, the first revision the actual ship, HMS Daring will be represented by a Product_as_realized related back to the Product_as_individual.

```
CLASS Product_as_individual_version;
SUBCLASS OF ( Product_version );
SUPERCLASS OF ( Product_as_realized, Product_as_planned, );
END_CLASS;
```

Class **Product as individual view** *[urn:plcs:rdl:std#Product_as_individual_view] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Product_as_individual_view is a type of Product_view_definition that defines a characterization view of a version of a Product_as_individual. NOTE: The Product_as_individual_view entity type supports the representation of different views of a Product_as_individual for different purposes. Multiple views of the same Product_as_individual are represented by different instances of Product_as_individual_view for the same Product_as_individual_version.

```
CLASS Product_as_individual_view;
SUBCLASS OF ( Product_view_definition );
END_CLASS;
```

Class **Product as planned** *[urn:plcs:rdl:std#Product_as_planned] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Product_as_planned is a type of Product_as_individual_version that identifies a revision of an individual artefact that has yet to be made. NOTE: It may be planned to make the artefact from of a version of a product design (Product_version). If this is the case, then the relationship between the artefact (Product_as_planned) and the design (Product_version) is represented by Product_design_version_to_individual. NOTE: If the planned artefact (Product_as_planned) is subsequently made into an actual artefact (Product_as_realized) then they are related by the Product_planned_to_realized relationship.

```
CLASS Product_as_planned;
SUBCLASS OF ( Product_as_individual_version );
END_CLASS;
```

Class **Product as realized** *[urn:plcs:rdl:std#Product_as_realized] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Product_as_realized is a type of Product_as_individual_version that identifies a revision of an individual artefact that has been made. A product whose properties can only be known by observation or by derivation from observations. NOTE: Where physical products are being represented, the Product_as_realized represents the physical product - something one can touch. NOTE: The artefact may have been made from a version of a product design (Product_version). If this is the case, then the relationship between the artefact (Product_as_realized) and the design (Product_version) is represented by Product_design_version_to_individual. NOTE: The artefact may have been planned and represented by Product_as_planned. In which case, the actual artefact (Product_as_realized) is related to the planned artefact (Product_as_planned) by the Product_planned_to_realized relationship.

```
CLASS Product_as_realized;
SUBCLASS OF ( Product_as_individual_version );
END_CLASS;
```

Class **Product based location identification** *[urn:plcs:rdl:std#Product_based_location_identification] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Product_based_location_identification is a type of Location_representation that specifies a location in the context of a product. EXAMPLE: Seat number M in the Aircraft.

```
CLASS Product_based_location_identification;
SUBCLASS OF ( Location_representation );
END_CLASS;
```

Class **Product category** *[urn:plcs:rdl:std#Product_category] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Product_category is a type of product that is defined for a purpose that is specific to a module or application protocol. NOTE: For the purpose of a general classification of products, use entity Class and Classification_assignment, standardized in ISO 10303-1070 and ISO 10303-1114.

```
CLASS Product_category;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Product concept** *[urn:plcs:rdl:std#Product_concept] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Product_concept is the identification of a set of similar products that were, are or will be proposed to customers. NOTE: The definition of product concepts is driven by market and customer requirements and forecasting. A Product_concept often corresponds to the highest level item(s) manufactured by an organization for customers. EXAMPLE: In an organization which manufactures cars and engines for cars, the car models will be represented by instances of Product_concept. NOTE: The entity data type Product_concept enables to represent customer-oriented identification of products that are to be delivered to customers, while the entity data type Product enables to identify and to track the history of items that are designed and manufactured, as a tangible solution, or component of the solution, for a product concept. NOTE: A product concept may be characterized by a set of product features identified by the customers or derived from customers' needs. NOTE: Depending on the kind of industry and products, a product concept might be offered to the customers in one or many different configurations.

```
CLASS Product_concept;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Product configuration** *[urn:plcs:rdl:std#Product_configuration] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Product_configuration is the identification of a Product_concept as a configuration. NOTE: The entity Product_configuration corresponds to the concept of configuration item defined, in some configuration management standards, as an item subject to configuration management. EXAMPLE: A Product_configuration may represent a component of a contracted product, onto which severe safety rules apply and for which configuration management is therefore rigorously applied. NOTE: A Product_configuration may identify a variation of a Product_concept, an entire Product_concept, or some portion thereof. NOTE: A Product_configuration can be established prior to the existence of a corresponding product.

```
CLASS Product_configuration;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Product design to individual** *[urn:plcs:rdl:std#Product_design_to_individual] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Product_design_to_individual is a relationship between a product design, represented by Product, and the product that is planned to be made or has been made from the design, represented by Product_as_individual.

```
CLASS Product_design_to_individual;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Product design version to individual**
[urn:plcs:rdl:std#Product_design_version_to_individual] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Product_design_version_to_individual is a relationship between a version of a product design, represented by Product_version, and the product that is planned to be made (Product_as_planned) or has been made (Product_as_realized) from the design.

```
CLASS Product_design_version_to_individual;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Product group** *[urn:plcs:rdl:std#Product_group] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Product_group is an identification of a set of Product_concept s, Product s, Product_group s, Product_version s or Product_as_individual s that have been grouped together. EXAMPLE: All the aircraft sold to BigPlanes airways.

```
CLASS Product_group;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Product group membership** *[urn:plcs:rdl:std#Product_group_membership] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Product_group_membership is an identification of an instance of an entity defined in the type product_select that belongs to a Product_group.

```
CLASS Product_group_membership;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Product group relationship** *[urn:plcs:rdl:std#Product_group_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Product_group_relationship is a relationship between two Product_group s. Classification_assignment is used to specify the meaning or type of the relationship. NOTE: A subset is a common type of relationship. It indicates that one Product_group is a subset of another.

```
CLASS Product_group_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Product in attachment slot** *[urn:plcs:rdl:std#Product_in_attachment_slot] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:
A Product_in_attachment_slot is a type of View_definition_usage that is a relationship between an Attachment_slot and a Product_view_definition of a Product that is designed to be attached to the attachment slot. EXAMPLE: A long-range fuel tank is designed to be attached to an aircraft in an attachment slot that corresponds to a pylon mounting on a wing.

```
CLASS Product_in_attachment_slot;
SUBCLASS OF ( View_definition_usage );
END_CLASS;
```

Class **Product planned to realized** *[urn:plcs:rdl:std#Product_planned_to_realized] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:
A Product_planned_to_realized is a relationship that establishes that a revision of a planned artefact represented by Product_as_planned has been realized as a revision of an actual artefact Product_as_realized.

```
CLASS Product_planned_to_realized;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Product relationship** *[urn:plcs:rdl:std#Product_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:
A Product_relationship is an association between two Product s.

```
CLASS Product_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Product version** *[urn:plcs:rdl:std#Product_version] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:
A Product_version is a revision of a Product. It is a collector of the definitions of this revision of the Product. NOTE: The set of all instances of Product_version of the same Product represents the history of the product.

```
CLASS Product_version;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Attachment_slot_version, Interface_specification_version,
Interface_connector_version, Breakdown_version, Product_as_individual_version,
Part_version, Breakdown_element_version, Document_version, Requirement_version,
);
END_CLASS;
```

Class **Product version relationship** *[urn:plcs:rdl:std#Product_version_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Product_version_relationship is an association between two versions of Product. NOTE: A relationship may exist between Product_version of different Product s or between different versions of the same Product.

```
CLASS Product_version_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Supplied_part_relationship, Requirement_version_relationship, );
END_CLASS;
```

Class **Product view definition** *[urn:plcs:rdl:std#Product_view_definition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Product_view_definition is a characterization of a Product_version, relevant in one or more application domains and for one or more life cycle stages. A Product_view_definition is a collector of the properties that characterize the Product_version in the initial_context and additional_contexts. EXAMPLE: The design of the SS Titanic and the as-built description of the SS Titanic can be represented as two instances of Product_view_definition.

```
CLASS Product_view_definition;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Requirement_view_definition, Part_view_definition,
Breakdown_element_definition, Attachment_slot_definition, Document_definition,
Interface_connector_definition, Product_as_individual_view,
Interface_specification_definition, );
END_CLASS;
```

Class **Project** *[urn:plcs:rdl:std#Project] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Project is an identified program of work.

```
CLASS Project;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Project assignment** *[urn:plcs:rdl:std#Project_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Project_assignment is a mechanism to associate a Project with activity or product data.

```
CLASS Project_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Project relationship** *[urn:plcs:rdl:std#Project_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Project_relationship is an association between two instances of Project with an identifiaction and a description of their relationship.

```
CLASS Project_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Promissory usage** *[urn:plcs:rdl:std#Promissory_usage] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Promissory_usage is a type of Assembly_component_relationship. It establishes a relationship between an assembly and a component, regardless of the number of intermediate levels between them, which may be established with instances of Next_assembly_usage. NOTE: A Promissory_usage may be used when the product structure is not completely defined, to capture the intent that the constituent will be used in that assembly.

```
CLASS Promissory_usage;
SUBCLASS OF ( Assembly_component_relationship );
END_CLASS;
```

Class **Property representation** *[urn:plcs:rdl:std#Property_representation] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Property_representation is an association between an Assigned_property and one of its Representation s.

```
CLASS Property_representation;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Property value representation** *[urn:plcs:rdl:std#Property_value_representation] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Property_value_representation is a type of Representation that represents one or more quantity values. NOTE: The present version of this application module does not enable to represent whether the values have been imposed or were derived or measured. It is expected that a future version will add this capability.

```
CLASS Property_value_representation;
SUBCLASS OF ( Representation );
SUPERCLASS OF ( Qualified_property_value_representation, );
END_CLASS;
```

Class **Qualification assignment** *[urn:plcs:rdl:std#Qualification_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Qualification_assignment is the assignment of a qualification to a person or organization.

```
CLASS Qualification_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Qualification type** *[urn:plcs:rdl:std#Qualification_type] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Qualification_type is the identification of a definitive recognition as a practitioner. EXAMPLE: A driving licence. EXAMPLE: A qualification for executing the Ground Running task for RB211 engines. EXAMPLE: A Military rank such as Colonel, or Captain EXAMPLE: Educational qualification such as GCSE, A level, Degree, Ordinary National Certificate, Higher National Certificate, City and Guilds, or GNVQ.

```
CLASS Qualification_type;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Qualification type relationship** *[urn:plcs:rdl:std#Qualification_type_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Qualification_type_relationship is an association between two Qualification_type s. EXAMPLE: Qualification X is a pre-requisite for qualification Y. EXAMPLE: Qualification X is equivalent to qualification Y.

```
CLASS Qualification_type_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Qualified property value representation** *[urn:plcs:rdl:std#Qualified_property_value_representation] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Qualified_property_value_representation is a type of Property_value_representation whose meaning or usage has been further qualified.

```
CLASS Qualified_property_value_representation;
SUBCLASS OF ( Property_value_representation );
END_CLASS;
```

Class **Random variable** *[urn:plcs:rdl:std#Random_variable] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Random_variable is a type of Numerical_item_with_global_unit that is the value of a random variable. NOTE: The value attribute is inherited from the supertype.

```
CLASS Random_variable;
SUBCLASS OF ( Numerical_item_with_global_unit );
END_CLASS;
```

Class **Ratio unit** *[urn:plcs:rdl:std#Ratio_unit] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:
A Ratio_unit is a type of Unit that expresses the dimensionless ratio of two quantities of the same kind.
CLASS **Ratio_unit**;
SUBCLASS OF (Unit);
END_CLASS;

Class **Regional coordinate** *[urn:plcs:rdl:std#Regional_coordinate] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:
A Regional_coordinate is a location that is specified relative to a Regional location system.
CLASS **Regional_coordinate**;
SUBCLASS OF (PLCS-ARM-LF-THING);
END_CLASS;

Class **Regional grid location representation** *[urn:plcs:rdl:std#Regional_grid_location_representation] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:
A Regional_grid_location_representation is a type of Location_representation that specifies a reference grid system.
CLASS **Regional_grid_location_representation**;
SUBCLASS OF (Location_representation);
END_CLASS;

Class **Related condition parameter** *[urn:plcs:rdl:std#Related_condition_parameter] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:
A Related_condition_parameter is a relationship between a Condition_parameter and a Condition_evaluation_parameter. This relationship is used to record the relationship between the parameters used to define a condition and the parameters used to evaluate it. EXAMPLE: The value of oil pressure (1.9 bar) used in Condition_evaluation (instance 87) was a measured value of the parameter used to define condition 29 (oil pressure on gauge 3).
CLASS **Related_condition_parameter**;
SUBCLASS OF (PLCS-ARM-LF-THING);
END_CLASS;

Class **Relative event** *[urn:plcs:rdl:std#Relative_event] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:
A Relative_event is a type of Event the start of which will occur, respectively has occurred, with a time offset from the start of another Event.
CLASS **Relative_event**;
SUBCLASS OF (Event);
END_CLASS;

Class **Repeat count** *[urn:plcs:rdl:std#Repeat_count] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Repeat_count is a type of Looping_element. It invokes a specified number of repetitions of the Looping_element.repeated_element Task_element inherited from the Looping_element supertype.

```
CLASS Repeat_count;
SUBCLASS OF ( Looping_element );
END_CLASS;
```

Class **Repeat until** *[urn:plcs:rdl:std#Repeat_until] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Repeat_until is a type of Looping_element. It invokes repetitions of a further Task_element and is repeated until the specified condition is satisfied. The element being repeated shall be executed at least once and the condition tested after the first execution.

```
CLASS Repeat_until;
SUBCLASS OF ( Looping_element );
END_CLASS;
```

Class **Repeat while** *[urn:plcs:rdl:std#Repeat_while] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Repeat_while is a type of Looping_element. It invokes repetitions of a further Task_element and is repeated while the specified condition is satisfied. The test condition shall be evaluated prior to invoking the method and may result in the Looping_element not being executed at all.

```
CLASS Repeat_while;
SUBCLASS OF ( Looping_element );
END_CLASS;
```

Class **Representation** *[urn:plcs:rdl:std#Representation] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Representation is a collection of one or more instances of Representation_item that are related in the specified Representation_context.

```
CLASS Representation;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Probability, Geometric_model, Document_property_representation,
Probability_generator, Property_value_representation, );
END_CLASS;
```

Class **Representation context** *[urn:plcs:rdl:std#Representation_context] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Representation_context is a context in which instances of Representation_item are related.

```
CLASS Representation_context;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Numerical_representation_context, );
END_CLASS;
```

Class **Representation item** *[urn:plcs:rdl:std#Representation_item] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Representation_item is an element of representation. A Representation_item shall be in the set of items of one or more instances of Representation or it shall belong to one or more instances of Representation, being referred to, directly of indirectly, by items of these instances of Representation. NOTE: this constraint is formally represented in the Express specification of the resource entity that corresponds to Representation_item in the MIM schema. Only specializations of Representation_item can be instantiated.

```
CLASS Representation_item;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Measure_item, Probability_named_value,
String_representation_item, Detailed_geometric_model_element, );
END_CLASS;
```

Class **Representation relationship** *[urn:plcs:rdl:std#Representation_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Representation_relationship is an association between two instances of Representation.

```
CLASS Representation_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Required resource** *[urn:plcs:rdl:std#Required_resource] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Required_resource is an identified need for resource. The role of a required resource is determined by classification. EXAMPLE: "facility", "test equipment", "supervisor" are examples of classifications of a required resource.

```
CLASS Required_resource;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Required_resource_by_resource_item,
Required_resource_by_specification, );
END_CLASS;
```

Class **Required resource assignment** *[urn:plcs:rdl:std#Required_resource_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Required_resource_assignment is an association of a resource requirement statement with one or more entities that requires the resource. EXAMPLE: task, task step, activity, activity method, organization are examples of entities to which the resource requirement statement could be related. The role of the assignment is determined by classification. EXAMPLE: The assignment can be classified as "required by". NOTE: An association between a required resource and actions that are needed prior to its usage. EXAMPLE: A resource required by the activity "12" needs to be calibrated prior to usage. The calibration activity "21" is associated with the same required resource.using EXAMPLE: A resource required by the task "123" needs to be disposed after its usage. This disposal task "456" is associated with the same required resource.

```
CLASS Required_resource_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Required resource by resource item**

[urn:plcs:rdl:std#Required_resource_by_resource_item] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Required_resource_by_resource_item is a type of Required_resource that is a statement of a required resource that can be identified within the application context.

```
CLASS Required_resource_by_resource_item;
SUBCLASS OF ( Required resource );
END_CLASS;
```

Class **Required resource by specification** *[urn:plcs:rdl:std#Required_resource_by_specification] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Required_resource_by_specification is a type of Required_resource where a collection of characteristics determine whether an item would be suitable to meet the requirement. More than one item can fulfill the requirement.

```
CLASS Required_resource_by_specification;
SUBCLASS OF ( Required resource );
END_CLASS;
```

Class **Required resource relationship** *[urn:plcs:rdl:std#Required_resource_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Required_resource_relationship is a relationship between two required resource statements. The meaning of the relationship is determined by classification. EXAMPLE: "alternative" and "realized by" are examples of required resource relationship. EXAMPLE: A Required_resource_by_specification for a power supply can be realized by a Required_resource_by_resource_item with part number "ABC-1234".

```
CLASS Required_resource_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Requirement** *[urn:plcs:rdl:std#Requirement] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Requirement is a type of Product that is used to uniquely identify a requirement. NOTE: The term "requirement" is used here in the sense that term is used in systems engineering and similar industrial domains. NOTE: There may be many versions of a requirement Requirement_version). There may also be more than one domain-specific view of a given Requirement_version (using the Requirement_view_definition entity). The requirement entity is simply a placeholder for holding a unique requirement. Most associations and properties are defined at the Requirement_view_definition level - that is in the context of a domain. EXAMPLE: A requirement may be identified as "NOx emissions requirement", and uniquely identified as "Req2". NOTE: Systems engineering tools and organizations may use differing identification mechanisms. Multiple identifiers may be assigned to a requirement using the Alias_identification entity.

```
CLASS Requirement;
SUBCLASS OF ( Product );
END_CLASS;
```

Class **Requirement assignment** *[urn:plcs:rdl:std#Requirement_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Requirement_assignment is used to relate a requirement (via the Requirement_view_definition entity) to data types representing the items which are affected by the requirement. EXAMPLE: A requirement "the vehicle shall have a maximum power output of at least 150BHP" could be assigned to the data types which are used to represent the vehicle's engine.

```
CLASS Requirement_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Requirement collection relationship** *[urn:plcs:rdl:std#Requirement_collection_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Requirement_collection_relationship is a type of View_definition_relationship that is used to relate a parent (collection) requirement to its member requirements. This provides a method for collecting together a set of requirements and treating them as a single requirement, whilst still being able to refer to individual requirements. NOTE: The inherited " View_definition_relationship.relating_view " and " View_definition_relationship.related_view " attributes have been renamed for purposes of clarity.

```
CLASS Requirement_collection_relationship;
SUBCLASS OF ( View_definition_relationship );
END_CLASS;
```

Class **Requirement source** *[urn:plcs:rdl:std#Requirement_source] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Requirement_source is a relationship between a requirement (via the Requirement_view_definition entity) and the data types representing the source of the

requirement EXAMPLE: The source of the requirement "the vehicle shall have a maximum power output of at least 150BHP" could be a document representing the findings of a market survey of sports car buyers.

```
CLASS Requirement_source;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Requirement version** *[urn:plcs:rdl:std#Requirement_version] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Requirement_version is a type of Product_version that identifies a particular version of a requirement. NOTE: This entity is used to record different versions of a requirement. In this case, the word "version" implies "revision" - that is a particular release of a requirement. EXAMPLE: A given requirement might have versions 1.0, 1.1, 1.2 and 2.0.

```
CLASS Requirement_version;
SUBCLASS OF ( Product_version );
END_CLASS;
```

Class **Requirement version relationship** *[urn:plcs:rdl:std#Requirement_version_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Requirement_version_relationship is a type of Product_version_relationship that is used to relate a previous version (predecessor) of a requirement to the version that replaces it (successor).

```
CLASS Requirement_version_relationship;
SUBCLASS OF ( Product_version_relationship );
END_CLASS;
```

Class **Requirement view definition** *[urn:plcs:rdl:std#Requirement_view_definition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Requirement_view_definition is a type of Product_view_definition that provides a view of a requirement version relevant for one or more application domains. This view collects requirement data for specific engineering purposes. EXAMPLE: An engineer may have responsibility for collecting all requirements associated with the cooling of an engine - covering engine block, tubing, water pump, electric fan. Some requirements in a Requirement_view_definition might impact on different disciplines.
Multiple Requirement_view_definition objects may be used to present different views of a given requirement for each discipline.

```
CLASS Requirement_view_definition;
SUBCLASS OF ( Product_view_definition );
END_CLASS;
```

Class **Resource as realized** *[urn:plcs:rdl:std#Resource_as_realized] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Resource_as_realized is a record of a resource that has been used or consumed. EXAMPLE: A resource as realized can be classified as "Used" or "Consumed".

```
CLASS Resource_as_realized;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Resource_as_realized_resource_item, );
END_CLASS;
```

Class **Resource as realized assignment** *[urn:plcs:rdl:std#Resource_as_realized_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Resource_as_realized_assignment is an association of a resource record statement with the entity that used or consumed the resource. The role of the assignment is determined by classification.

```
CLASS Resource_as_realized_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Resource as realized relationship** *[urn:plcs:rdl:std#Resource_as_realized_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Resource_as_realized_relationship is a relationship between the record of used or consumed resources and the corresponding statement of Required_resource. EXAMPLE: "Alternative" and "Corresponding" are examples of classifications of resource as realized relationship. NOTE: A resource used or consumed can be recorded without having a corresponding resource requirement statement.

```
CLASS Resource_as_realized_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Resource as realized resource item** *[urn:plcs:rdl:std#Resource_as_realized_resource_item] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Resource_as_realized_resource_item is a type of Resource_as_realized that is a used or consumed resource.

```
CLASS Resource_as_realized_resource_item;
SUBCLASS OF ( Resource_as_realized );
END_CLASS;
```

Class **Resource event** *[urn:plcs:rdl:std#Resource_event] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Resource_event is an event or action that affects the balance or availability of a managed resource. The role of a resource event is determined by classification. EXAMPLE: A resource event can be classified as "Planned" or "Actual".

```
CLASS Resource_event;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Increasing_resource_event, Decreasing_resource_event, );
END_CLASS;
```

Class **Resource event correspondence relationship**

[urn:plcs:rdl:std#Resource_event_correspondence_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Resource_event_correspondence_relationship is an association of a resource event and a corresponding statement of Required_resource. The meaning of the relationship is determined by classification. NOTE: A resource event can be planned or recorded without having a corresponding resource requirement statement. EXAMPLE: A resource event correspondence relationship can be classified as "Designated for".

```
CLASS Resource_event_correspondence_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Resource event relationship** *[urn:plcs:rdl:std#Resource_event_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Resource_event_relationship is a specification of how an Resource_event may be associated with another Resource_event. The meaning of the relationship is determined by classification. EXAMPLE: A managed resource relationship can be classified as "realized by" or "replaces".

```
CLASS Resource_event_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Resource group relationship** *[urn:plcs:rdl:std#Resource_group_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Resource_group_relationship is a type of Resource_item_relationship that specifiies the means to associate two resource items that are part of a resource group. The meaning of the entity is determined by classification. EXAMPLE: The relationship between a tool set and a mallet could be classified as "Contains". EXAMPLE: The relationship between a facility and compressed air could be classified as "Provides".

```
CLASS Resource_group_relationship;
SUBCLASS OF ( Resource_item_relationship );
END_CLASS;
```

Class **Resource item** *[urn:plcs:rdl:std#Resource_item] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Resource_item is an item that can occur in the role of a resource within the application context. EXAMPLE: A Resource_item may be classified as "Facility", "Replaceble unit", or "Package".

```
CLASS Resource_item;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Resource item assignment** *[urn:plcs:rdl:std#Resource_item_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Resource_item_assignment is an association of a resource item with some product information. The role of the assignment is determined by classification. EXAMPLE: The assignment can be classified as "applicable to" or "acquired for".

```
CLASS Resource_item_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Resource item relationship** *[urn:plcs:rdl:std#Resource_item_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Resource_item_relationship is a specification of how a resource item may be associated with another resource item. The role of the relationship is determined by classification. EXAMPLE: The relationship between a product and a package can be classified as "Preferred".

```
CLASS Resource_item_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Resource_group_relationship, );
END_CLASS;
```

Class **Resource property** *[urn:plcs:rdl:std#Resource_property] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Resource_property is a property of a resource related object. NOTE: The resource related object is specified in another module

```
CLASS Resource_property;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Applied_independent_resource_property, );
END_CLASS;
```

Class **Resource property representation** *[urn:plcs:rdl:std#Resource_property_representation] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Resource_property_representation is an association between an Resource_property and one of its representations.

```
CLASS Resource_property_representation;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Scheme** *[urn:plcs:rdl:std#Scheme] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:
A Scheme is a type of Activity_method. It provides the identification and description of an intended course of action to accomplish an objective. A Scheme enables the ordering of entries. Dates and times may be specified for entries and time intervals between entries. NOTE: A Scheme may be classified as a Plan or Schedule, and it may be further classified into specific types of Plans or Schedules. EXAMPLE: Acquisition plan, Maintenance plan, Resource schedule are examples of schemes.

```
CLASS Scheme;
SUBCLASS OF ( Activity_method );
END_CLASS;
```

Class Scheme entry *[urn:plcs:rdl:std#Scheme_entry] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:
A Scheme_entry is a type of Activity_method that provides the identification and description of a single entry in a specific Scheme. NOTE: A Scheme_entry may be associated with time constraints. NOTE: A Scheme_entry only exists within the scope of a specific Scheme.

```
CLASS Scheme_entry;
SUBCLASS OF ( Activity_method );
END_CLASS;
```

Class Scheme entry assignment *[urn:plcs:rdl:std#Scheme_entry_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:
A Scheme_entry_assignment is a type of Applied_activity_method_assignment. It associates a Scheme_entry with one of more items. NOTE: The Scheme_entry_assignment links the single items included in Plans and Schedules with their associated Scheme_entry. These items may be actions, events, or tasks depending on the nature of the Plan or Schedule.

```
CLASS Scheme_entry_assignment;
SUBCLASS OF ( Applied_activity_method_assignment );
END_CLASS;
```

Class Scheme entry relationship *[urn:plcs:rdl:std#Scheme_entry_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:
A Scheme_entry_relationship is a type of Activity_method_relationship. It relates two Scheme_entry entities. An association may exists between Scheme_entry entities that relate to different Scheme or between different Scheme_entry entity instances for the same Scheme. NOTE: The Scheme_entry_relationship provides the ability to relate entries included in Plans or Schedules in different ways. By applying classifications on the Scheme_entry_relationship it can be used for different purposes. EXAMPLE: Decomposition, Dependency, and sequencing are examples of kinds of relationships possible between schema entries.

```
CLASS Scheme_entry_relationship;
SUBCLASS OF ( Activity_method_relationship );
SUPERCLASS OF ( Sequencing_relationship, );
END_CLASS;
```

Class **Scheme relationship** *[urn:plcs:rdl:std#Scheme_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Scheme_relationship is a type of Activity_method_relationship relating two Scheme s. NOTE: The Scheme_relationship provides the ability to relate Plans or Schedules represented by the Scheme entity in different ways. If classifications are available to the schema using this on, by applying classifications on the Scheme_relationship it can be used for different purposes. EXAMPLE: Decomposition, based-on, alternative, version are kinds of relationships between Scheme s.

```
CLASS Scheme_relationship;
SUBCLASS OF ( Activity_method_relationship );
END_CLASS;
```

Class **Scheme subject assignment** *[urn:plcs:rdl:std#Scheme_subject_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Scheme_subject_assignment is a type of Applied_activity_method_assignment. It associates a Scheme with a specific subject. NOTE: The Scheme_subject_assignment links the Plans and Schedules with their associated subjects or targets. This may indicate the intent of the scheme. EXAMPLE: The maintenance plan for an individual vehicle, where the subject attribute points to an entity instance representing the individual vehicle.

```
CLASS Scheme_subject_assignment;
SUBCLASS OF ( Applied_activity_method_assignment );
END_CLASS;
```

Class **Scheme version** *[urn:plcs:rdl:std#Scheme_version] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Scheme_version is a type of Activity_method. It enables version control of Scheme.

```
CLASS Scheme_version;
SUBCLASS OF ( Activity_method );
END_CLASS;
```

Class **Scheme version assignment** *[urn:plcs:rdl:std#Scheme_version_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Scheme_version_assignment is a type of Applied_activity_method_assignment. It associates a Scheme with a information describing the version. NOTE: The role of the association may be defined through classification.

```
CLASS Scheme_version_assignment;
SUBCLASS OF ( Applied_activity_method_assignment );
END_CLASS;
```

Class **Scheme version relationship** *[urn:plcs:rdl:std#Scheme_version_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Scheme_version_relationship is a type of Activity_method_relationship relating two Scheme_version s. NOTE: The meaning of the relationship may be defined through classification.

```
CLASS Scheme_version_relationship;
SUBCLASS OF ( Activity_method_relationship );
END_CLASS;
```

Class **Security classification** *[urn:plcs:rdl:std#Security_classification] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Security_classification is a level of confidentiality that can be applied to protect activity or product data against unauthorized usage.

```
CLASS Security_classification;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Security classification assignment** *[urn:plcs:rdl:std#Security_classification_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Security_classification_assignment is an association of a Security_classification with activity or product data.

```
CLASS Security_classification_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Selected item** *[urn:plcs:rdl:std#Selected_item] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Selected_item is a type of Class used to classify identified product or activity data as being significant for configuration management purposes. EXAMPLE: Time Change/Shelf Life Selected Item (TCSI/SLSI)is an example of a selected item - All items that must be replaced at specific intervals or that have an elapsed time-related shelf life, are designated as TCSIs or SLSIs, as appropriate. These items do not require the rigorous, up-front CM activity (for example FCA/PCA but rather, they concentrate on the post-delivery CM aspects related to traceability in the field. This is commensurate with their emphasis on being changed/replaced at certain times. EXAMPLE: Maintenance Selected Item (MSI) is an example of a selected item - Certain items whose maintenance requirements dictate a more detailed traceability than normal items, may be designated as MSIs. MSIs emphasize post-delivery traceability for maintenance purposes. EXAMPLE: Support Equipment Selected Item (SESI) is an example of a selected item - Certain support equipment items that require some additional traceability, but do not require up-front CM activity are designated as SESIs. These items typically require traceability by lot or batch, but not by individual item serial number.

```
CLASS Selected_item;
SUBCLASS OF ( Class );
END_CLASS;
```

Class **Selected item assignment** *[urn:plcs:rdl:std#Selected_item_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Selected_item_assignment is the identification of the product or activity data referenced by Selected_item_assignment.item as being a Selected_item.

```
CLASS Selected_item_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Sequence of state** *[urn:plcs:rdl:std#Sequence_of_state] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Sequence_of_state is a type of State_relationship in which one set of State precedes another set of State.

```
CLASS Sequence_of_state;
SUBCLASS OF ( State_relationship );
END_CLASS;
```

Class **Sequence of state definition** *[urn:plcs:rdl:std#Sequence_of_state_definition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Sequence_of_state_definition is a type of State_definition_relationship. It defines a sequence of two or more state definitions.

```
CLASS Sequence_of_state_definition;
SUBCLASS OF ( State_definition_relationship );
END_CLASS;
```

Class **Sequencing relationship** *[urn:plcs:rdl:std#Sequencing_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Sequencing_relationship is a type of Scheme_entry_relationship. It defines a specific type of sequencing and relative timing for two Scheme_entry. NOTE: Specific types of sequencing could include start-start, finish-start.

```
CLASS Sequencing_relationship;
SUBCLASS OF ( Scheme_entry_relationship );
END_CLASS;
```

Class **Serial effectivity** *[urn:plcs:rdl:std#Serial_effectivity] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Serial_effectivity is a type of Effectivity for which the domain of applicability is defined as a possibly open-ended interval of serial numbers.

CLASS **Serial_effectivity**;
SUBCLASS OF (Effectivity);
END_CLASS;

Class **Simultaneous elements** *[urn:plcs:rdl:std#Simultaneous_elements] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Simultaneous_elements is a type of Concurrent_elements that represents two or more actions to be performed together.

CLASS **Simultaneous_elements**;
SUBCLASS OF (Concurrent_elements);
END_CLASS;

Class **Solid angle unit** *[urn:plcs:rdl:std#Solid_angle_unit] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Solid_angle_unit is a type of Unit that is solid angle.

CLASS **Solid_angle_unit**;
SUBCLASS OF (Unit);
END_CLASS;

Class **State** *[urn:plcs:rdl:std#State] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A State is the mode of being in which something does or could exist or existed for a period of time. NOTE: A state's existence can be just a state that an object is currently in, a predicted state that an object will eventually be in, or an observed state that an object has been in. NOTE: The period of existence may be an instant or longer. EXAMPLE: Main Engine No. 1 is in "operation". EXAMPLE: When Generator No. 2 surpasses 5,000 service hours, it will enter "maintenance" mode. EXAMPLE: The portable computer's power supply was attached after it displayed a "low-battery" warning.

CLASS **State**;
SUBCLASS OF (PLCS-ARM-LF-THING);
SUPERCLASS OF (State_predicted, State_observed,);
END_CLASS;

Class **State assertion** *[urn:plcs:rdl:std#State_assertion] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A State_assertion is used to assert that the subject State is in conformance with a particular State_definition.

CLASS **State_assertion**;
SUBCLASS OF (PLCS-ARM-LF-THING);
END_CLASS;

Class **State assessment** *[urn:plcs:rdl:std#State_assessment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A State_assessment is used to determine whether the subject State is comparable with a particular State_definition.

```
CLASS State_assessment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **State cause effect** *[urn:plcs:rdl:std#State_cause_effect] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A State_cause_effect is a type of state relationship and it relates two or more states as one state causing particular resulting effect state(s). In turn, an effect state can become a new causing state yielding in yet more effect states.

```
CLASS State_cause_effect;
SUBCLASS OF ( State_relationship );
END_CLASS;
```

Class **State cause effect definition** *[urn:plcs:rdl:std#State_cause_effect_definition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A State_cause_effect_definition is a type of State_definition_relationship that is used to define a causal relationship between two sets of State_definition entities. At least one State_definition acts as a cause and at least one State_definition acts as a effect. NOTE: Additional causal relationships between states can be expressed using the following subtypes: And_state_cause_effect_definition, Or_state_cause_effect_definition, and Xor_state_cause_effect_definition.

```
CLASS State_cause_effect_definition;
SUBCLASS OF ( State_definition_relationship );
SUPERCLASS OF ( Xor_state_cause_effect_definition,
And_state_cause_effect_definition, Or_state_cause_effect_definition, );
END_CLASS;
```

Class **State complement definition** *[urn:plcs:rdl:std#State_complement_definition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A State_complement_definition is a type of State_definition_relationship. It is a relationship among three sets of State_definition entities. It defines the complement of a set of State_definition entities relative to a set of State_definition entities that are the universe. NOTE: The semantics are the same as in elementary set theory. NOTE: The relationship between a State_definition and its complement is symmetrical.

```
CLASS State_complement_definition;
SUBCLASS OF ( State_definition_relationship );
END_CLASS;
```

Class **State definition** *[urn:plcs:rdl:std#State_definition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A State_definition is a mode of being. A State_definition defines the types of State that can exist. A State 's existence can be assessed and asserted against State_definition from State.

```
CLASS State_definition;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **State definition relationship** *[urn:plcs:rdl:std#State_definition_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A State_definition_relationship is an association between two or more instances of State_definition. NOTE: Relationships between State_definition entities may be used to support fault diagnosis.

```
CLASS State_definition_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( State_transition_definition, Composition_of_state_definition,
State_proper_subset_definition, State_cause_effect_definition,
State_subset_definition, Sequence_of_state_definition,
State_complement_definition, State_symptom_definition, );
END_CLASS;
```

Class **State observed** *[urn:plcs:rdl:std#State_observed] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A State_observed is a type of State. It is an individual or realized State that is observed.

```
CLASS State_observed;
SUBCLASS OF ( State );
END_CLASS;
```

Class **State predicted** *[urn:plcs:rdl:std#State_predicted] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A State_predicted is a type of State It is a predicted actual State. Where a predicted state needs to be related to an observed state, the State_predicted_to_observed entity shall be used.

```
CLASS State_predicted;
SUBCLASS OF ( State );
END_CLASS;
```

Class **State predicted to observed** *[urn:plcs:rdl:std#State_predicted_to_observed] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:
A State_predicted_to_observed is a type of state relationship. It specifies the relationship between two individual states, one of which is a State_predicted to a second state which is a State_observed.
CLASS **State_predicted_to_observed;**
SUBCLASS OF (State_relationship);
END_CLASS;

Class State proper subset definition *[urn:plcs:rdl:std#State_proper_subset_definition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:
A State_proper_subset_definition is a type of State_definition_relationship. It is a relationship between two sets of State_definition entities. NOTE: The relationship between a state and its environment can be described as a State_proper_subset_definition. The identification of an instrinsic state is the proper_subset. The identification of an extrinsic state is the proper_superset.
CLASS **State_proper_subset_definition;**
SUBCLASS OF (State_definition_relationship);
END_CLASS;

Class State relationship *[urn:plcs:rdl:std#State_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:
A State_relationship is a relationship between two or more State.
CLASS **State_relationship;**
SUBCLASS OF (PLCS-ARM-LF-THING);
SUPERCLASS OF (State_transition, State_predicted_to_observed, Composition_of_state, Sequence_of_state, State_cause_effect,);
END_CLASS;

Class State role *[urn:plcs:rdl:std#State_role] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:
A State_role is a specification of the purpose of the association of the Applied_state_assignment with product or activity data.
CLASS **State_role;**
SUBCLASS OF (PLCS-ARM-LF-THING);
END_CLASS;

Class State subset definition *[urn:plcs:rdl:std#State_subset_definition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:
A State_subset_definition is a type of State_definition_relationship. It is a relationship between two sets of State_definition entities. NOTE: The first set may be equal to the second set.
CLASS **State_subset_definition;**
SUBCLASS OF (State_definition_relationship);
END_CLASS;

Class **State symptom definition** *[urn:plcs:rdl:std#State_symptom_definition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A State_symptom_definition is a type of State_definition_relationship. It relates two or more State_definition entities in regards to symptom, where a symptom is something that indicates the existence of something else. At least one State_definition acts as a symptom_cause and at least one State_definition acts as a symptom_effect.

```
CLASS State_symptom_definition;
SUBCLASS OF ( State_definition_relationship );
END_CLASS;
```

Class **State transition** *[urn:plcs:rdl:std#State_transition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A State_transition is a type of state relationship and it relates two or more states before and after a transition in State, where at least one State is a start state and at least one State is an end state.

```
CLASS State_transition;
SUBCLASS OF ( State_relationship );
END_CLASS;
```

Class **State transition definition** *[urn:plcs:rdl:std#State_transition_definition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A State_transition_definition is a type of State_definition_relationship. It relates two or more State_definition entities before and after a transition in state, where at least one State_definition is a start_state and at least one State_definition is an end_state.

```
CLASS State_transition_definition;
SUBCLASS OF ( State_definition_relationship );
END_CLASS;
```

Class **String representation item** *[urn:plcs:rdl:std#String_representation_item] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A String_representation_item is a type of Representation_item that specifies a text.

```
CLASS String_representation_item;
SUBCLASS OF ( Representation_item );
SUPERCLASS OF ( Descriptive_document_property, );
END_CLASS;
```

Class **Structured task element** *[urn:plcs:rdl:std#Structured_task_element] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Structured_task_element is a type of Task_element. It is made up of other Task_element s
CLASS **Structured_task_element**;
SUBCLASS OF (Task_element);
SUPERCLASS OF (Task_element_sequence, Looping_element, Decision_point, Concurrent_elements,);
END_CLASS;

Class **Supplied part relationship** *[urn:plcs:rdl:std#Supplied_part_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:
A Supplied_part_relationship is a type of Product_version_relationship that relates two instances of Product_version that represent the same object in different organizational contexts. One of the organizations is the supplier of the object to the other organization. This entity is applicable for part versions and document versions. NOTE: This entity enables to represent the fact that two organizations may use distinct identifiers to identify their Product s and their versions. NOTE: This mechanism can only be used in an information system or in exchange files where the content of the id attribute of instances of Product is not constrained by a particular identification scheme. NOTE: The module provides a more general mechanism that can be used to track alias identifiers for any entity data type that has a id attribute.
CLASS **Supplied_part_relationship**;
SUBCLASS OF (Product_version_relationship);
END_CLASS;

Class **System breakdown** *[urn:plcs:rdl:std#System_breakdown] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:
A System_breakdown is a type of Breakdown that identifies a partitioning of a system into a set of related elements so as to form explicit, assembly - component views that comprise the system elements. The assembly-component view is represented by System_element_usage instances relating the system elements in the breakdown which are represented by System_element s. EXAMPLE: A system breakdown provides a decomposition of an aircraft in terms of high-level mechanisms such as fuel system or flight control system - which might, in the second example, further decompose into low-level systems such as autopilot system and instrument landing system.
CLASS **System_breakdown**;
SUBCLASS OF (Breakdown);
END_CLASS;

Class **System breakdown context** *[urn:plcs:rdl:std#System_breakdown_context] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:
A System_breakdown_context is a type of Breakdown_context that is a membership relationship between a System_element and a System_breakdown of which the system element is a member. EXAMPLE: A heating system is a member of the breakdown of a climate control system.
CLASS **System_breakdown_context**;
SUBCLASS OF (Breakdown_context);
END_CLASS;

Class **System breakdown version** *[urn:plcs:rdl:std#System_breakdown_version] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A System_breakdown_version is a type of Breakdown_version that identifies a version of a System_breakdown. EXAMPLE: A logistics engineer modifies the current systems breakdown for an aircraft and associated support equipment on the basis of results from a level of repair analysis.

```
CLASS System_breakdown_version;
SUBCLASS OF ( Breakdown_version );
END_CLASS;
```

Class **System element** *[urn:plcs:rdl:std#System_element] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A System_element is a type of Breakdown_element that identifies the elements in one or more System_breakdown objects.

```
CLASS System_element;
SUBCLASS OF ( Breakdown_element );
END_CLASS;
```

Class **System element definition** *[urn:plcs:rdl:std#System_element_definition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A System_element_definition is a type of Breakdown_element_definition that identifies a view of a version (System_element_version) of a System_element. EXAMPLE: The collision avoidance system element of a system breakdown is subject to a level of repair analysis to support implementation of optimized maintenance for an aircraft.

```
CLASS System_element_definition;
SUBCLASS OF ( Breakdown_element_definition );
END_CLASS;
```

Class **System element usage** *[urn:plcs:rdl:std#System_element_usage] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A System_element_usage is a type of Breakdown_element_usage that is a relationship between a System_element_definition and another System_element_definition that is a constituent. EXAMPLE: In a system breakdown, the fuel system might include a fuel storage system and a fuel injection system as components.

```
CLASS System_element_usage;
SUBCLASS OF ( Breakdown_element_usage );
END_CLASS;
```

Class **System element version** *[urn:plcs:rdl:std#System_element_version] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A System_element_version is a type of Breakdown_element_version that identifies a version of a System_element. EXAMPLE: A sound engineer changes the details describing the public address system that is an element in a system breakdown of an aircraft.

```
CLASS System_element_version;
SUBCLASS OF ( Breakdown_element_version );
END_CLASS;
```

Class **Task element** *[urn:plcs:rdl:std#Task_element] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Task_element is a type of Activity_method. It is a representation of all or part of how to undertake a task.

```
CLASS Task_element;
SUBCLASS OF ( Activity_method );
SUPERCLASS OF ( Task_element_levels, Task_invocation, Structured_task_element,
Exit_loop, Task_step, End_task, );
END_CLASS;
```

Class **Task element assignment** *[urn:plcs:rdl:std#Task_element_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Task_element_assignment is a type of Applied_activity_method_assignment. It is an association of a Task_element with product or activity data.

```
CLASS Task_element_assignment;
SUBCLASS OF ( Applied_activity_method_assignment );
END_CLASS;
```

Class **Task element levels** *[urn:plcs:rdl:std#Task_element_levels] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Task_element_levels is a type of Task_element that provides two or more different descriptions in place of a single method. The actual work will be the same whichever alternative Task_element is followed. NOTE: This can be used to provide defferent levels of description of a task for people with varying levels of experience or expertise.

```
CLASS Task_element_levels;
SUBCLASS OF ( Task_element );
END_CLASS;
```

Class **Task element relationship** *[urn:plcs:rdl:std#Task_element_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Task_element_relationship is a type of Activity_method_relationship. It relates two instances of Task_element. EXAMPLE: Can be used to capture a time dependency that cuts across the structure of the method.

```
CLASS Task_element_relationship;
SUBCLASS OF ( Activity_method_relationship );
SUPERCLASS OF ( Element_constraint, );
END_CLASS;
```

Class **Task element sequence** *[urn:plcs:rdl:std#Task_element_sequence] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Task_element_sequence is a type of Task_element that comprises a sequence of steps to be followed in a specified order.

```
CLASS Task_element_sequence;
SUBCLASS OF ( Structured_task_element );
END_CLASS;
```

Class **Task element state relationship** *[urn:plcs:rdl:std#Task_element_state_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Task_element_state_relationship is an type of Activity_method_relationship. It is an association between a State or a State_definition and a Task_element. The meaning of the entity is determined by classification. Candidate meanings include: * assumed starting state; * required starting state.

```
CLASS Task_element_state_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Task invocation** *[urn:plcs:rdl:std#Task_invocation] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Task_invocation is a type of Task_element. It is an instruction to perform another task. EXAMPLE: A Task_element calls a pre-defined task to perform an instrument calibration.

```
CLASS Task_invocation;
SUBCLASS OF ( Task_element );
END_CLASS;
```

Class **Task method** *[urn:plcs:rdl:std#Task_method] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Task_method is a type of Activity_method. It is a specification of work. NOTE: The task method may be implemented using people, machines or a combination.

```
CLASS Task_method;
SUBCLASS OF ( Activity_method );
END_CLASS;
```

Class **Task method assignment** *[urn:plcs:rdl:std#Task_method_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Task_method_assignment is a type of Applied_activity_method_assignment. It is an association of a Task_method with product or activity data.

```
CLASS Task_method_assignment;
SUBCLASS OF ( Applied_activity_method_assignment );
END_CLASS;
```

Class **Task method relationship** *[urn:plcs:rdl:std#Task_method_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Task_method_relationship is a type of Activity_method_relationship. It relates two task methods. NOTE: The nature of the relationship is determined from its classification.

```
CLASS Task_method_relationship;
SUBCLASS OF ( Activity_method_relationship );
END_CLASS;
```

Class **Task method state relationship** *[urn:plcs:rdl:std#Task_method_state_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Task_method_state_relationship is a relationship between a state and a Task_method. NOTE: The meaning of the entity is determined by classification. Candidate meanings include: * Assumed starting state; * Required starting state.

```
CLASS Task_method_state_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Task method version** *[urn:plcs:rdl:std#Task_method_version] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Task_method_version is a type of Activity_method which allows the explicit reference to versions of a Task_method and to track changes against a Task_method.

```
CLASS Task_method_version;
SUBCLASS OF ( Activity_method );
END_CLASS;
```

Class **Task method version assignment** *[urn:plcs:rdl:std#Task_method_version_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Task_method_version_assignment is a type of Applied_activity_method_assignment. It is an association of a Task_method_version with product or activity data.

```
CLASS Task_method_version_assignment;
SUBCLASS OF ( Applied_activity_method_assignment );
END_CLASS;
```

Class **Task method version relationship** *[urn:plcs:rdl:std#Task_method_version_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Task_method_version_relationship is a type of Activity_method_relationship. It relates two instances of Task_method_version.

```
CLASS Task_method_version_relationship;
SUBCLASS OF ( Activity_method_relationship );
END_CLASS;
```

Class **Task objective** *[urn:plcs:rdl:std#Task_objective] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Task_objective is a result or objective that is reached by undertaking a Task_method. NOTE: More than one Task_method may be defined for a given objective, if there can be several ways to accomplish the objective.

```
CLASS Task_objective;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Task objective state relationship** *[urn:plcs:rdl:std#Task_objective_state_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Task_objective_state_relationship is a relationship between a state and a Task_objective. NOTE: The meaning of the entity is determined by classification. Candidate meanings include: * Intended finishing state; * Alternative finishing state.

```
CLASS Task_objective_state_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Task step** *[urn:plcs:rdl:std#Task_step] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Task_step is a type of Task_element that is not further sub-divided.

```
CLASS Task_step;
SUBCLASS OF ( Task_element );
SUPERCLASS OF ( Advisory_task_step, );
END_CLASS;
```

Class **Thermodynamic temperature unit** *[urn:plcs:rdl:std#Thermodynamic_temperature_unit] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Thermodynamic_temperature_unit is a type of Unit in which the degree of heat of a body is expressed.

```
CLASS Thermodynamic_temperature_unit;
SUBCLASS OF ( Unit );
END_CLASS;
```

Class **Time interval effectivity** *[urn:plcs:rdl:std#Time_interval_effectivity] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Time_interval_effectivity is a type of Effectivity for which the domain of applicability is defined as a Time_interval.

```
CLASS Time_interval_effectivity;
SUBCLASS OF ( Effectivity );
END_CLASS;
```

Class **Time interval relationship** *[urn:plcs:rdl:std#Time_interval_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Time_interval_relationship is the association of two instances of Time_interval. The meaning of this association is specified in the Time_interval_relationship.relation_type attribute.

```
CLASS Time_interval_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Time unit** *[urn:plcs:rdl:std#Time_unit] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Time_unit is a type of Unit with which the duration of a period is expressed.

```
CLASS Time_unit;
SUBCLASS OF ( Unit );
END_CLASS;
```

Class **Tracing relationship** *[urn:plcs:rdl:std#Tracing_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Tracing_relationship is a type of View_definition_relationship that shows tracing from (Tracing_relationship.traces_from) one requirement to another (Tracing_relationship.traces_to). EXAMPLE: A requirement on the performance of a catalytic converter in a car may be traced from a more general emissions requirement. A requirement may trace to many other requirements and vice versa - this is achieved by creating multiple instances of the tracing relationship entity. NOTE: The inherited " View_definition_relationship.relating_view " and " View_definition_relationship.related_view " attributes have been renamed for purposes of

clarity. NOTE: Properties may be attached to tracing relationships. This is intended to deal with "user defined" attributes which are common on tracing relationships in requirements tools.

```
CLASS Tracing_relationship;
SUBCLASS OF ( View_definition_relationship );
END_CLASS;
```

Class Transformation based template instance

[urn:plcs:rdl:std#Transformation_based_template_instance] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Transformation_based_template_instance is a type of Detailed_geometric_model_element. It is the replication of a Geometric_model. The replication transformation is defined by a cartesian_transformation. NOTE: In the case where the Geometric_coordinate_space of the replicated model is not the same as the Geometric_coordinate_space in which the template_instance is founded, care shall be taken of any difference in the length units of both the instances of Geometric_coordinate_space, as unit conversion may be required. NOTE: The mapping-table defines two interpretations for this ARM entity. The first one uses the MIM entity mapped_item which uses the same principle to define the transformation as the Axis_placement_mapping and Axis_placement_transformation_mapping entities. The second one uses the MIM entity representation_relationship_with_transformation. Although this latter entity is not semantically equivalent to the ARM entity, the second interpretation is kept because of its use in some implementations.

```
CLASS Transformation_based_template_instance;
SUBCLASS OF ( Detailed_geometric_model_element );
END_CLASS;
```

Class Type of person *[urn:plcs:rdl:std#Type_of_person] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Type_of_person is a type of person. EXAMPLE: Class 3 welder EXAMPLE: Truck driver EXAMPLE: Electronics engineer EXAMPLE: Mechanical technician EXAMPLE: Helicopter pilot

```
CLASS Type_of_person;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class Type of person assignment *[urn:plcs:rdl:std#Type_of_person_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Type_of_person_assignment is the association of a Type_of_person in a role with an activity or product data.

```
CLASS Type_of_person_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class Type of person definition *[urn:plcs:rdl:std#Type_of_person_definition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Type_of_person_definition is the definition of a Type_of_person in terms of required properties or attributes. EXAMPLE: A junior mechanical design engineer could be specified to be either someone who has * 3 years experience of working in a mechanical design department, or * a degree in mechanical engineering.

```
CLASS Type_of_person_definition;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class Type of person definition relationship

[urn:plcs:rdl:std#Type_of_person_definition_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Type_of_person_definition_relationship is a relationship between two definitions of a type of person (Type_of_person_definition). EXAMPLE: "alternate" and "superceded by" are examples of Type_of_person_definition_relationship s. NOTE: The meaning of the relationship is determined by classification.

```
CLASS Type_of_person_definition_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class Type of person definition required attributes relationship

[urn:plcs:rdl:std#Type_of_person_definition_required_attributes_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Type_of_person_definition_required_attributes_relationship is a relationship between a Type_of_person_definition and the attributes required to define that type of person. EXAMPLE: The type of person "van driver" is required to possess the qualification named "commercial driving license" or the experience level "3 years of driving more than 10,000 miles per year".

```
CLASS Type_of_person_definition_required_attributes_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class Uncertainty with unit *[urn:plcs:rdl:std#Uncertainty_with_unit] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Uncertainty_with_unit is a type of Value_with_unit that specifies the uncertainty that applies to a type of measure. An Uncertainty_with_unit applies to each Representation_item that uses the type of measure specified in the Value_with_unit.value_component of the Uncertainty_with_unit.

```
CLASS Uncertainty_with_unit;
SUBCLASS OF ( Value_with_unit );
END_CLASS;
```

Class Unit *[urn:plcs:rdl:std#Unit] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Unit is a unit quantity.

```
CLASS Unit;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Conversion_based_unit, Context_dependent_unit, Solid_angle_unit,
Amount_of_substance_unit, Derived_unit, Luminous_intensity_unit, Time_unit,
Ratio_unit, Plane_angle_unit, Length_unit, Electric_current_unit,
Thermodynamic_temperature_unit, Mass_unit, );
END_CLASS;
```

Class **Value limit** *[urn:plcs:rdl:std#Value_limit] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Value_limit is a type of Measure_item that specifies a qualified numerical value representing either the lower limit or the upper limit of a particular quantifiable characteristic.

```
CLASS Value_limit;
SUBCLASS OF ( Measure_item );
END_CLASS;
```

Class **Value limit with global unit** *[urn:plcs:rdl:std#Value_limit_with_global_unit] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Value_limit_with_global_unit is a type of Measure_item that specifies a qualified numerical value representing either the lower limit or the upper limit of a particular quantifiable characteristic and for which the value unit is globally assigned.

```
CLASS Value_limit_with_global_unit;
SUBCLASS OF ( Measure_item );
END_CLASS;
```

Class **Value list** *[urn:plcs:rdl:std#Value_list] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Value_list is a type of Measure_item that is an ordered collection of Measure_item s. EXAMPLE: A Measure_item may be composed of different values such as 'mass', 'speed', and 'age' which are all necessary in a given context. The Value_list collects all of them in a given order, such that each is identifiable by its index in the list.

```
CLASS Value_list;
SUBCLASS OF ( Measure_item );
END_CLASS;
```

Class **Value range** *[urn:plcs:rdl:std#Value_range] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:
A Value_range is a type of Measure_item that is a pair of numbers representing the range in which the value shall lie.
```
CLASS Value_range;
SUBCLASS OF ( Measure_item );
END_CLASS;
```

Class **Value range with global unit** *[urn:plcs:rdl:std#Value_range_with_global_unit] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:
A Value_range_with_global_unit is a type of Measure_item that is a pair of numerical values representing the range in which the value shall lie. The unit of measure is globally assigned.
```
CLASS Value_range_with_global_unit;
SUBCLASS OF ( Measure_item );
END_CLASS;
```

Class **Value set** *[urn:plcs:rdl:std#Value_set] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:
A Value_set is a type of Measure_item that is an unordered collection of Measure_item s. EXAMPLE: A Measure_item may be composed of different values such as 'mass', 'speed', and 'age' which are all necessary in a given context. The Value_set collects all of them in a given order, such that each is identifiable by its index in the list.
```
CLASS Value_set;
SUBCLASS OF ( Measure_item );
END_CLASS;
```

Class **Value with tolerances** *[urn:plcs:rdl:std#Value_with_tolerances] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:
A Value_with_tolerances is a type of Measure_item that specifies a range of values by specifying a single nominal value and two tolerances that are offsets from the single value. The range is defined to be the closed interval [item value + lower limit, item value + upper limit].
```
CLASS Value_with_tolerances;
SUBCLASS OF ( Measure_item );
END_CLASS;
```

Class **Value with unit** *[urn:plcs:rdl:std#Value_with_unit] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:
A Value_with_unit is the specification of a physical quantity by its value and its unit.
```
CLASS Value_with_unit;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Uncertainty_with_unit, Duration, Numerical_item_with_unit, );
END_CLASS;
```

Class **View definition context** *[urn:plcs:rdl:std#View_definition_context] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A View_definition_context is the grouping of an application domain and a life cycle stage. It identifies a universe of discourse suitable for the description of products. NOTE: Requirements and vocabularies vary among the industrial activity fields. This entity intends to identify such a domain.

```
CLASS View_definition_context;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **View definition relationship** *[urn:plcs:rdl:std#View_definition_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A View_definition_relationship is an association between two instances of Product_view_definition. This association represents a relationship between the product versions, indirectly identified by the instances of Product_view_definition, relevant in the definition contexts of the related instances of Product_view_definition.

```
CLASS View_definition_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
SUPERCLASS OF ( Tracing_relationship, View_definition_usage,
Requirement_collection_relationship, );
END_CLASS;
```

Class **View definition usage** *[urn:plcs:rdl:std#View_definition_usage] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A View_definition_usage is a type of View_definition_relationship that specifies a directed association between two instances of Product_view_definition. This association represents a relationship stating that, in the definition contexts of the related instances of Product_view_definition, it is considered that the related product is used in the context of the relating product.

```
CLASS View_definition_usage;
SUBCLASS OF ( View_definition_relationship );
SUPERCLASS OF ( Assembly_component_relationship, Breakdown_element_usage,
Product_in_attachment_slot, Make_from_relationship, );
END_CLASS;
```

Class **Work order** *[urn:plcs:rdl:std#Work_order] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Work_order is an authoritative instrument which provides directions to achieve the specified results. A Work_order is the authorization for one or more Activity objects to be performed.

```
CLASS Work_order;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Work output** *[urn:plcs:rdl:std#Work_output] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Work_output is a statement of output resulting from an activity or a task. The role of the work output is determined by classification. EXAMPLE: When applicable a work output may be classified as * "Planned"; * "Actual"; * "By-product"; * "Waste product".

```
CLASS Work_output;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Work output assignment** *[urn:plcs:rdl:std#Work_output_assignment] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Work_output_assignment is an association of a work output statement with the source that produces or delivers the output. The work output can be planned as well as actual. The role of the assignment is determined by classification.

```
CLASS Work_output_assignment;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Work output relationship** *[urn:plcs:rdl:std#Work_output_relationship] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Work_output_relationship is a relationship between two work output statements. The meaning of the relationship is determined by classification. EXAMPLE: A Work_output_relationship is classified as "realized by".

```
CLASS Work_output_relationship;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Work request** *[urn:plcs:rdl:std#Work_request] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Work_request is the solicitation for some work to be done. NOTE: These requests may not be acted upon depending on the authorization granted to the request or its associated Work_order.

```
CLASS Work_request;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class **Work request status** *[urn:plcs:rdl:std#Work_request_status] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Work_request_status is an association of a status with a Work_request. NOTE: A Work_request may have zero or more statuses, assigned at various dates by various organizations.

```
CLASS Work_request_status;
SUBCLASS OF ( PLCS-ARM-LF-THING );
END_CLASS;
```

Class Xor state cause effect definition *[urn:plcs:rdl:std#Xor_state_cause_effect_definition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

An Xor_state_cause_effect_definition is a type of State_cause_effect_definition. It relates one of the single or many causing state definition(s) and one effect State_definition, whereby any and only one of the causing state definitions exists prior to the single effect to take place.

```
CLASS Xor_state_cause_effect_definition;
SUBCLASS OF ( State_cause_effect_definition );
END_CLASS;
```

Class Zone breakdown *[urn:plcs:rdl:std#Zone_breakdown] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Zone_breakdown is a type of Breakdown that identifies a partitioning of a product into a set of related zonal elements so as to form explicit, parent-child views that comprise the product elements. The parent-child view is represented by Zone_element_usage instances relating the zonal elements in the breakdown which are represented by Zone_element s. EXAMPLE: A zonal breakdown provides a means of identifying the decomposition of an aircraft in terms of spaces or high-level conceptual parts such as 'wing' - which might further decompose into lower-level zones such as 'inner-wing', and 'outer wing'.

```
CLASS Zone_breakdown;
SUBCLASS OF ( Breakdown );
END_CLASS;
```

Class Zone breakdown context *[urn:plcs:rdl:std#Zone_breakdown_context] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Zone_breakdown_context is a type of Breakdown_context that is a membership relationship between a Zone_element and a Zone_breakdown of which the zonal element is a member. EXAMPLE: A 'fire-check zone' might be a member of the zonal breakdown of a building.

```
CLASS Zone_breakdown_context;
SUBCLASS OF ( Breakdown_context );
END_CLASS;
```

Class Zone breakdown version *[urn:plcs:rdl:std#Zone_breakdown_version] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Zone_breakdown_version is a type of Breakdown_version that identifies a version of a Zone_breakdown. EXAMPLE: An architect modifies the current fire-check zone breakdown for an building on the basis of reports from a buildings inspector.

```
CLASS Zone_breakdown_version;
SUBCLASS OF ( Breakdown_version );
END_CLASS;
```

Class **Zone element** *[urn:plcs:rdl:std#Zone_element] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Zone_element is a type of Breakdown_element that identifies the elements in one or more Zone_breakdown objects. EXAMPLE: 'Empennage', 'Right vertical stabilizer and rudder' and 'Lower rudder' are all elements in a zonal breakdown of an aircraft.

```
CLASS Zone_element;
SUBCLASS OF ( Breakdown_element );
END_CLASS;
```

Class **Zone element definition** *[urn:plcs:rdl:std#Zone_element_definition] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Zone_element_definition is a type of Breakdown_element_definition that identifies a view of a version (Zone_element_version) of a Zone_element. EXAMPLE: For an aircraft, an element 'Right vertical stabilizer tip' is in a zonal breakdown that an engineer uses for reliability-centred maintenance analysis.

```
CLASS Zone_element_definition;
SUBCLASS OF ( Breakdown_element_definition );
END_CLASS;
```

Class **Zone element usage** *[urn:plcs:rdl:std#Zone_element_usage] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Zone_element_usage is a type of Breakdown_element_usage that is a relationship between a parent and child Zone_element. EXAMPLE: In a zonal breakdown, the 'wing' (parent) might include (as children) an 'inner wing' and an 'outer wing'.

```
CLASS Zone_element_usage;
SUBCLASS OF ( Breakdown_element_usage );
END_CLASS;
```

Class **Zone element version** *[urn:plcs:rdl:std#Zone_element_version] [file: committee-specification/plcs-arm-lf-express.rdf-xml.owl]*

Source:ISO 10303-239 **Creator:**OASIS PLCS Technical Committee Reference Data Development Team

Definition:

A Zone_element_version is a type of Breakdown_element_version that identifies a version of a Zone_element. EXAMPLE: An engineer defines an inspection task on a breakdown element 'Upper rudder' that is part of a zonal breakdown of an aircraft. The engineer identifies the corresponding view of the breakdown element.

```
CLASS Zone_element_version;
SUBCLASS OF ( Breakdown_element_version );
END_CLASS;
```

Annex IV: ALA Functional View Data Dictionary

This dictionary has been defined by AleniaAermacchi with the support of the National research initiative named **iDesign Foundation**.

Entity Name	Description	Attributes
Function	Functions are discrete actions necessary to achieve the system's objectives. The functions will ultimately be performed or accomplished through use of equipment, personnel, facilities, software, or a combination.	**Name**: name of the function.
		Description: textual description of the function objectives.
		Level: a function can be divided in two or more sub functions, so this attribute points out the hierarchical functional level.
		Input: resource external to a function that is transformed in order to reach the desired result.
		Output: the result produced by the function starting from its inputs.
Functional Requirement	A functional/performance requirement is a set of attributes or documents that specify characteristics to be satisfied in product design. It specifies the parent entity System Requirement.	It inherits all the attributes of the partent entity System Requirement
Functional System Specification	A specification is a technical document that describes at a different level of definition (system, subsystem or equipment level) how to design the product satisfying a set of requirements.	**Filing_Code**: it traces how the document has been stored.
		Title: title of the document.
		Program: it is the name of aircraft program the requirements refer to.
Interface	The interface refers to the point of interaction between different system elements.	**Name**: name of the interface.
		Type: it is a Boolean attribute, it can be 'physical' or 'logical'.
		Description: textual description of interface characteristics.

Entity Name	Description	Attributes
Logical System Architecture	It is a combination of interacting elements (e.g. system elements, interface) organized to achieve one or more stated purposes.	**Architecture_Code**: it univocally identifies the architecture.
		Name: name of the architecture.
		Description: textual description of the architecture characteristics.
Non-Functional Requirement	A non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors (e.g. usability, testability, maintainability, extensibility and scalability). It specifies the parent entity System Requirement.	It inherits all the attributes of the partent entity System Requirement
Parameter	Definable, measurable, and constant or variable characteristic, dimension, property, or value, because it is considered essential to understanding a situation (or in solving a problem).	**Name**: name of the parameter.
		Description: textual description of the parameter characteristics.
		Value: the numerical value that the parameter can assume.
Requirement Specification	It is a technical document that gathers all the system requirements.	**Filing_Code**: it traces how the document has been stored.
		Title: title of the document.
		Program: it is the name of aircraft program the requirements refer to.
System Element	It is a member of a set of elements that constitutes a system.	**System_Breackdown_Number**: it corresponds to the system/subsystem decomposition of the product with a specific code number associated to each system/subsystem. This numbering system is a prefix to be used to identify requirements, specifications, parts, etc.
		ATA_Code: the code assigned according to the international standard ATA (Air Transportation Association).
		NATO_Code: the code assigned according to the international standard

Entity Name	Description	Attributes
		NATO (North Atlantic Treaty Organization).
		Name: name of the system element.
		Description: textual description of the system element characteristics.
		Level: a system element can be divided in two or more sub system elements, so this attribute points out the hierarchical level.
		Type: typology of the system element (e.g. assembly, component, part).
System Requirement	A statement that identifies a system characteristic or constraint, which is unambiguous, clear, unique, consistent, stand-alone (not grouped), and verifiable, and is deemed necessary for stakeholder acceptability.	**Requirement_Code**: it is allocated, according to the system breakdown numbering and it univocally identifies the requirement.
		Name: name of the requirement.
		Description: textual description of the requirement object.
		Version: the requirement can be modified, so this attribute points out the related version.
		State: it is a Boolean value, and it can be 'verified' or 'not verified'.
		Type: typology of the requirement. It can be "Performance Requirement", "Reference Requirement", "Phisical Property Requirement", "Interface Requirement", "Effectiveness Measure Requirement".
Trade-off Analysis	Decision-making actions that select from various requirements and alternative solutions on the basis of net benefit to the stakeholders.	**Name**: name of the analysis.
		Description: textual description of the analysis characteristics.
		Result: the outcome of the analysis activities to choose the best solution.
		Input: information necessary for the

Entity Name	Description	Attributes
		analysis process.
Validation	The process of evaluating during or at the end of the system development process to determine whether the requirements for a specific intended use or application have been fulfilled [ISO 9000: 2000]	**Validation_Code**: it univocally identifies the validation activity.
		Name: name of the validation acitivity.
		Description: textual description of the validation activity characteristics.
		Result: the outcome of the validation activities.
		Tool: the instrument used to perform the validation activity.
		Type: the typology of validation activity performed (e.g. MoC)
		Procedure: sequence of activities composing the validation.

Annex V: BDA package overview