Date: August 2016

Unified Architecture Framework Profile (UAFP)

Version 1.0 – FTF Beta 1

OMG Document Number: dtc/16-08-01

Standard document URL: http://www.omg.org/spec/UAF/1.0

Normative Machine Consumable File(s):

http://www.omg.org/spec/UAF/20160505/UAFP_Profile.xmi

http://www.omg.org/spec/UAF/20160506/Class-Library-UAF

This OMG document replaces the submission document (c4i/2016-05-01). It is an OMG Adopted Beta specification and is currently in the finalization phase. Comments on the content of this document are welcome, and should be entered by September 27, 2016 using the Issue Reporting Form on the main web page http://www.omg.org, under Documents, Report a Bug/Issue (http://issues.omg.org/issues/create-new-issue).

The FTF Recommendation and Report for this specification will be published on June 16, 2017. If you are reading this after that date, please download the available specification from the OMG Specifications Catalog.

USE OF SPECIFICATION - TERMS, CONDITIONS & NOTICES

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

LICENSES

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

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

PATENTS

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

GENERAL USE RESTRICTIONS

Any unauthorized use of this specification may violate copyright laws, trademark laws, and communications regulations and statutes. This document contains information which is protected by copyright. All Rights Reserved. No part of this work covered by copyright herein may be reproduced or used in any form or by any

DISCLAIMER OF WARRANTY

RESTRICTED RIGHTS LEGEND

TRADEMARKS

COMPLIANCE

Table of Contents

TABLE OF FIGURES

Preface

OMG

Founded in 1989, the Object Management Group, Inc. (OMG) is an open membership, not-for-profit computer industry standards consortium that produces and maintains computer industry specifications for interoperable, portable and reusable enterprise applications in distributed, heterogeneous environments. Membership includes Information Technology vendors, end users, government agencies and academia. OMG member companies write, adopt, and maintain its specifications following a mature, open process. OMG's specifications implement the Model Driven Architecture® (MDA®), maximizing ROI through a full-lifecycle approach to enterprise integration that covers multiple operating systems, programming languages, middleware and networking infrastructures, and software development environments. OMG's specifications include: UML® (Unified Modeling Language™); CORBA® (Common Object Request Broker Architecture); CWM™ (Common Warehouse Metamodel); and industry-specific standards for dozens of vertical markets. More information on the OMG is available at *http://www.omg.org/*.

OMG Specifications

As noted, OMG specifications address middleware, modeling and vertical domain frameworks. All OMG Specifications are available from this URL: *http://www.omg.org/spec*

Specifications are organized by the following categories:

Business Modeling Specifications

Middleware Specifications

- CORBA/IIOP
- Data Distribution Services
- Specialized CORBA IDL/Language Mapping Specifications

Modeling and Metadata Specifications

- UML, MOF, CWM, XMI
- UML Profile Specifications

Platform Independent Model (PIM) - Platform Specific Model (PSM) - Interface Specifications

- CORBAServices
- CORBAFacilities
- OMG Domain Specifications
- CORBA Embedded Intelligence Specifications
- CORBA Security Specifications

All of OMG's formal specifications may be downloaded without charge from our website. (Products implementing OMG specifications are available from individual suppliers.) Copies of specifications, available in PostScript and PDF format, may be obtained from the Specifications Catalog cited above or by contacting the Object Management Group, Inc. at: OMG Headquarters 109 Highland Avenue, Needham, MA 02494 USA Tel: +1- 781-444-0404 Fax: +1-781-444-0320 Email: *pubs@omg.org*

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

Typographical Conventions

The type styles shown below are used in this document to distinguish programming statements from ordinary English. However, these conventions are not used in tables or section headings where no distinction is necessary.

Times/Times New Roman - 10 pt.: Standard body text

Helvetica/Arial - 10 pt. Bold: OMG Interface Definition Language (OMG IDL) and syntax elements.

`Courier - 10 pt. Bold`: Programming language elements.

Helvetica/Arial - 10 pt: Exceptions

Note – Terms that appear in *italics* are defined in the glossary. Italic text also represents the name of a document, specification, or other publication.

Issues

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

1. Scope

1.1 UAFP Background

The scope of Unified Architecture Framework Profile (UAFP) includes the language extensions to enable the extraction of specified and custom models from an integrated architecture description (AD). The models describe a system[1] from a set of stakeholders' concerns such as security or information through a set of predefined viewpoints and associated views[2]. Developed models can also reflect custom viewpoints or to develop more formal extensions for new viewpoints. The UAFP specification supports the Department of Defense Architecture Framework (DoDAF) 2.02, the Ministry of Defence Architecture Framework (MODAF), Security Views from Canada's Department of National Defense Architecture Framework (DNDAF) and the North Atlantic Treaty Organization (NATO) Architecture Framework (NAF) v 3.1. The core concepts in the UAF domain metamodel specify the UAFP based upon the DoDAF 2.0.2 Domain Metamodel (DM2) and the MODAF ontological data exchange mechanism (MODEM). MODEM is intended to provide the basis for the next version of NAF). The intent is to provide a standard representation for AD support for Defense Organizations. The intention of UAFP is also to support a standard representation for non-defense organizations' ADs as part of their Systems Engineering (SE) technical processes. The associated UAF metamodel (see C4i-2016-02-03) intent is to improve the ability to exchange architecture data between related tools that are UML/SysML based and tools that are based on other standards.

UAFP v 1.0 supports the capability to:

- model architectures for a broad range of complex systems, which may include hardware, software, data, personnel, and facility elements;
- model consistent architectures for system-of-systems (SoS) down to lower levels of design and implementation;
- support the analysis, specification, design, and verification of complex systems; and
- improve the ability to exchange architecture information among related tools that are SysML based and tools that are based on other standards.

1.2 Intended Users

The profile enables the modeling of strategic capabilities; business/operational activities, nodes and their interfaces, measures of effectiveness; services and their interfaces, levels of agreement and measures of performance; system resources and their functions, ports, protocols, interfaces, measures of performance; security including cyber security controls; human interactions with systems to support business operations; information and data schemas; and project planning. In addition, the profile enables the modeling of related architecture concepts such as System of Systems (SoS), information exchanges consistent with the National Information Exchange Model (NIEM), DoD's doctrine, organization, training material, leadership & education, personnel, and facilities (DOTMLPF) and the equivalent UK Ministry of Defence Lines of Development (DLOD) elements, classification markings, business processes, and Human Computer Interfaces (HCI).

Further, The profile conforms to terms defined in the ISO/IEC/IEEE 42010 standard for architecture description, where the terms: *architecture*, *architecture description (AD)*, *architecture framework*, *architecture view*, *architecture viewpoint*, *concern*, *environment*, *model kind*, *stakeholder* [ISO/IEC/IEEE 42010:2011] form correspondence rules specified as constraints on UAFP.

This document specifies a SysML v1.3 profile to enable practitioners to express architectural model elements and organize them in a set of specified viewpoints and views that support the specific needs of systems engineers in the defense and manufacturing industries. At least four tool vendors including IBM, No Magic, PTC, and Sparx Systems will

[1] The term system is used from: "Systems and software engineering -- Architecture description," http://www.iso.org/iso/catalogue_detail.htm?csnumber=50508

[2] Stakeholder, concern, viewpoint, view and model are also used from "Systems and software engineering -- Architecture description," http://www.iso.org/iso/catalogue_detail.htm?csnumber=50508

support the specifications defined profile. The vendors plan to release a commercially available product supporting the revised version of UAFP within next year. Currently, implementations of the predecessor profile, UPDM v 2.1, are actively used on projects.

UAFP 1.0 defines a set of SysML stereotypes and model elements and associations to satisfy the requirements of the UPDM 3.0 RFP[3]. The profile specification documents the language architecture in terms of the parts of SysML that are reused and the defined extensions to SysML. The specification includes the concrete syntax (notation) for the complete language. The reusable portions of the SysML specification are not included directly in the specification but are made through reference.

1.3 Related Documents

The specification includes a metatmodel and description as separate documents. Other appendicies are also provided as separate documents. The table below provides a listing of these documents:

Table 1. Table of Related Documents

c4i/16-05-01	The UAFP Specification
c4i/16-05-02	Appendix A the UAF domain metamodel
c4i/16-05-03	Appendix B that contains a separate traceability subsection from UAFP to each of the frameworks listed in Section 1.1 of this specification
c4i/16-05-04	Appendix C: An example of how the language can be used to represent an UAFP architecture
c4i/16-05-05	UAF XMI file
c4i/16-05-06	UAF XMI class library
c4i/16-05-07	Inventory file

[3] http://www.omg.org/cgi-bin/doc.cgi?c4i/2013-9-11

2. Conformance

UAFP 1.0 specifies one level of compliance corresponding to supporting a SysML™ profile using SysML v 1.3. UAFP imports the SysML profile and defines constraints that pair together the application of SysML and UAFP stereotypes. This provides a UAFP implementation that seamlessly evolves forward into SysML modeling. For a tool to be considered as compliant with UAFP, the following must be true:

- All stereotypes, classes, attributes, constraints, associations and package structures must exist and becompliant with this specification.
- XMI import and export of the user model and profile must be supported.
- An UAFP compliant implementation must be able to import and export UAFP models with 100% fidelity (i.e., no loss or transforms).

3. References

3.1 Normative References

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

3.2 OMG Documents (Normative References)

- Unified Modeling Language (UML), 2.5, June 2015, http://www.omg.org/spec/UML
- Object Constraint Language (OCL), 2.4, February 2014, http://www.omg.org/spec/OCL
- System Modeling Language (SysML) ,1.4, September 2015, http://www.omg.org/spec/SysML
- Diagram Definition (DD), 1.1, June 2015, http://www.omg.org/spec/DD
- UML Profile for the National Information Exchange Model (NIEM UML), 1.0, June 2014, http://www.omg.org/spec/NIEM-UML
- Unified Profile for DoDAF and MODAF (UPDM), 2.1, August 2013, http://www.omg.org/spec/UPDM
- UML Profile for BPMN Processes, 1.0, July 2014, http://www.omg.org/spec/BPMNProfile
- Ontology Definition Metamodel (ODM), 1.1, September 2014, http://www.omg.org/spec/ODM
- Information Exchange Packaging Policy Vocabulary (IEPPV) 1.0, May 2015, http://www.omg.org/spec/IEPPV

3.3 Other Normative References

- Department of Defense Architecture Framework (DoDAF), Version 2.02, August 2010, http://dodcio.defense.gov/Library/DoDArchitectureFramework.aspx
- DM2 - DoDAF Meta-Model,
- The DM2 Conceptual Data Model, http://dodcio.defense.gov/Library/DoDArchitectureFramework/dodaf20_conceptual.aspx
- DM2 Logical Data Model, http://dodcio.defense.gov/Library/DoDArchitectureFramework/dodaf20_logical.aspx
- DM2 Formal Ontology. http://dodcio.defense.gov/Library/DoDArchitectureFramework/dodaf20_ontology1.aspx
- Department National Defence and Canadian Forces (DND/ CF) Architecture Framework (DNDAF), Version 1.8.1, 25 January 2013
- International Defence Enterprise Architecture Specification for Exchange (IDEAS) Group, http://www.ideasgroup.org/
- IDEAS Foundation, http://www.ideasgroup.org/foundation/
- IDEAS Foundation v1.0 as XMI File (zipped), http://www.ideasgroup.org/7Documents/
- ISO/IEC/IEEE 42010:2011, Systems and software engineering – Architecture Description, http://www.iso.org/iso/catalogue_detail.htm?csnumber=50508
- Ministry of Defence Architecture Framework (MODAF), https://www.gov.uk/mod-architecture-framework
- MODAF Ontological Data Exchange Mechanism (MODEM)
- https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/63980/20130117_MODAF_MODEM.pdf
- NATO Architecture Framework (NAF),
- Version 3, NATO C3 BOARD (AC/322-D(2007)0048), http://www.nhqc3s.nato.int/HomePage.asp (no longer available publicly available online as of 3 November 2015)
- NATO Architecture Framework v4.0 Documentation is in draft. Available from http://nafdocs.org/

3.4 Informative References

- Business Process Model & Notation (BPMN), Version 2.0.2, December 2013 http://www.omg.org/spec/BPMN
- ISO 15704:2000, Industrial Automation Systems – "Requirements for Enterprise-Reference Architectures and Methodologies," http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=28777
- ISO 8601:2004 Data elements and interchange formats – Information interchange – Representation of dates and times, http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?ics1=01&ics2=140&ics3=30&csnumber=40874
- ISO/IEC 15288:2015, "Systems Engineering - Systems Life Cycle Processes," http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=63711
- Object Management Group (OMG), Metamodel Extension Facility, Initial submission, ad/12-02-01, http://www.omg.org/cgi-bin/doc?ad/12-02-01 (Requires OMG Member Access)
- OASIS SOA-RAF, Reference Architecture Foundation for Service Oriented Architecture Version 1.0, OASIS SOA Reference Model TC, 04 December 2012. http://docs.oasis-open.org/soa-rm/soa-ra/v1.0/cs01/soa-ra-v1.0-cs01.pdf (Authoritative)
- Object Management Group (OMG), Semantics of Business Vocabulary and Business Rules (SBVR), Version 1.3, May 2015, http://www.omg.org/spec/SBVR
- Business Motivation Model (BMM), Version 1.3, http://www.omg.org/spec/BMM/1.3/
- International Council On Systems Engineering (INCOSE), Systems Engineering Handbook V4, 2015, http://www.incose.org/ProductsPublications/sehandbook

4. Terms and Definitions

No new terms and definitions have been required to create this specification. All terms are available in the normative references or bibliographic citations for detailed explanation.

5. Symbols

For the purposes of this specification, the following List of symbols/abbreviations apply.

AcV-*[4]	Acquisition View
AD	Architecture Description
AV-*	All View
BMM	Business Motivation Model
BPMN	Business Process Modeling Notation
C4ISR	Command, Control, Communications, Computers, Intelligence, Surveillance, and Reconnaissance
CaT	Capability Team
COI	Communities of Interest
CV-*	Capability View
DIV-*	Data and Information Views
DLOD	Defence Lines of Development
DM2	DoDAF Meta Model
DMM	Domain Meta Model
DNDAF	Department National Defence and Canadian Forces (DND/ CF) Architecture Framework
DoD	United States Department of Defense
DoDAF	Department of Defense Architecture Framework
DOTMLP	Doctrine, Organization, Training, Material, Leadership, Personnel, Facilities
EIE	Enterprise Information Environment
IDEAS	International Defense Enterprise Architecture Specification for Exchange
IDEF	Integrated DEFinition Methods
INCOSE	International Council Of Systems Engineering
JCIDS	Joint Capabilities Integration and Development System
MISIG	Model Interchange Special Interest Group
MOD	United Kingdom Ministry of Defence
MODAF	Ministry of Defence Architecture Framework
MODEM	MODAF Ontological Data Exchange Mechanism
NAF	NATO Architecture Framework
OASIS	Organization for the Advancement of Structured Information Standards
OSLC	Open Services for Lifecycle Collaboration
OV-*	Operational View
PES	DoDAF Physical Exchange Specification
POC	Proof of Concept
PV-*	Project View
RDF	Resource Description Framework
SoaML	Service orientated architecture Modeling Language
SoS	System of Systems
SOV-*	Service Oriented View
StdV-*	Standards View in DoDAF 2.02 compare TV-* in UAF

[4] * denotes a wildcard

STV-*	Strategic View
SV-*	System View
SvcV-*	Service View
TEPID OIL	Training, Equipment, Personnel, Information, Concepts and Doctrine, Organisation, Infrastructure, Logistics
TOGAF	The Open Group Architectural Framework©
TPPU	Task, Post, Process, and Use
TV-*	Technical View
UAF	Unified Architecture Framework
UAFP	Unified Architecture Framework Profile
UPDM	Unified Profile for DoDAF/MODAF

6. Additional Information

6.1 Changes to Adopted OMG Specifications

This specification completely replaces Unified Profile for DoDAF and MODAF (UPDM), Version 2.1, August 2013, http://www.omg.org/spec/UPDM/2.1, and it supersedes the UPDM v2.2. The issues reported for UPDM v 2.2 FTF are resolved and subsumed in this UAFP 1.0 specification.

6.2 Language Architecture

The UAFP specification reuses a subset of UML 2 and SysML 1.3 and provides additional extensions needed to address requirements in the UPDM 3.0 RFP Mandatory Requirements. Those requirements form the basis for this specification. This specification documents the language architecture in terms of the UML 2 and SysML 1.3 parts that are reused and the defined UML 2 extensions; and specifies how to implement UAFP. This clause explains design principles and how they are applied to define the UAFP language architecture.

6.3 Philosophy

The UAFP development uses a model-driven approach. A simple description of the work process is:

- A Domain Metamodel (DMM) uses UML Class models to represent the concepts in DoDAF, MODEM, NAF, DNDAF and the other contributing frameworks.
- The aligned and unified concepts from these frameworks provides a common domain metamodel usable by all the contributing frameworks thereby separating the metamodel from the existing definition of presentation layer in the contributing framework.
- The aligned and renamed viewpoints from the various frameworks provide a common generic name for each viewpoint. It should be noted that the term viewpoint is in the context of ISO 42010 where a viewpoint is the specification of a view. The UAFP viewpoints are mapped to the corresponding viewpoint in the relevant contributing framework. It is the viewpoints described in the DMM that provides the basis for the Unified Architecture Framework (UAF).
- The UAF provides an abstraction layer that separates the underlying UAF from the presentation layer. The results of this mapping are given in Appendix B (see document c4i/16-05-03), and an overview of the viewpoints in a grid format are given in Appendix B, Annex A.
- The intent of the UAF is provide a Domain MetaModel usable by non UML/SysML tool vendors who may wish to implement the UAF within their own tool and metalanguage. It is unneccassary to generate XMI for the UAF as toolvendors basing their implementation on the UAF are unlikely to support XMI.
- The Unified Architecture Profile (UAFP) is derivable from the DMM (UAF) by mapping the UAF concepts and relationships to corresponding stereotypes in the UAFP.
- The UAFP analyzsis and refactoring reflects language architecture, tool implementation, and reuse considerations.
- The UAFP diagrams, stereotype descriptions, and documentation are added.
- The specification is generated from the UAFP model.

This approach allows the team to concentrate on architecture issues rather than documentation production. The UML tool automatically maintains consistency.

The UML tool improves maintainance and enables traceability between the UAFP and the UAF where every stereotype is linkable to the UAF element using UML Abstraction relationship.

There are two key parts to this submission:

1. A UAF (Appendix A, see document c4i/16-05-02) providing the domain meta-model and viewpoints for the framework. This enables non-UML tool vendors implementions.

2. A UAF Profile (this document) for UML/SysML derivable from the UAF that specifies how UML/SysML tool vendors should implement the profile.

The intent from this two-document approach is to make the specification practical to implement for both UML/SysML and non-UML/SysML tool vendors.

The DMM and profile definitions enable SysML tool vendors to perform behavioral analysis using simulation; and the evaluation of non-functional requirements using parametric diagram execution and analysis. For implementers of non-SysML tools, the hope is that they can achieve similar types of analysis using proprietary technology.

The expectaton is implementers of this specification should follow the view naming conventions of the framework they are intending to implement based on this specification.

6.4 Core Principles

The fundamental design principles for UAFP are:

- **Requirements-driven**: UAFP is intended to satisfy the requirements of the UPDM 3.0 RFP Mandatory Requirements.
- **Domain meta model (DMM) driven**: The DMM was created first by domain experts and it served as a foundation for profile development.
- **Reuse of existing specifications**: UAFP reuses UML/SysML wherever practical to satisfy the requirements of the UAFP 3.0 RFP and leverage features from both UML and SysML to provide a robust modeling capability. Consequently, UAFP is intended to be relatively easy to implement for vendors who support UML 2 and SysML.
- **Partitioning**: The package is the basic unit of partitioning in this specification. The packages partition the model elements into logical groupings that minimize circular dependencies among them.
- **Compliance levels**: UAFP has a single compliance level based upon a combination of the reuse of UML and SysML elements, this simplifies the implementation of UAFP compared to UPDM 2.x for tool vendors. It is expected that the views that are created as result of this profile have frames that reflect the underlying SysML diagram type that is used as the basis for the view. It also expected that the graphical notation used to display elements within those views correspond to the standard SysML graphical notation of the SysML/UML metaclass that the stereotype extends.
- **Interoperability**: UAFP inherits the XMI interchange capability from UML. The UAFP specification reuses a subset of UML 2 and provides additional extensions needed to address requirements in the UPDM 3.0 RFP Mandatory Requirements. We have used those requirements as the basis for this specification. This specification documents the language architecture in terms of the parts of UML 2 and SyML 1.4 and the respective extensions that are used to implement the UAFP.

6.5 Representing Stereotype Constraints

The UAF Profile uses an enhanced standard notation to represent metaconstraints graphically in the UAF profile diagrams to improve readability of the UAF Profile specification and overcome limitations of being unable to visualize constraints diagrammatically in UML.

The metaconstraints appears in the UAFP specification diagrams for visualization purposes only, however the represention in the XMI is as a UML constraint, specified in structured English. These constraints are implementable in a tool, by OCL for example.

A simple UML profile defines these metaconstraints.

The following subsections detail the metaconstraint profile definition within the UAF profile.

6.5.1 Metaconstraint dependency

«metaconstraint» is a stereotype that extends the Dependency metaclass. It is used to specify constrained elements within the profile.

A sample of the «metaconstraint» dependency is a diagram for stereotype extending the Dependency metaclass.

MapsToCapability is a UAFP stereotype that extends Abstraction (a type of Dependency in UML). The constraint on this stereotype is that its client end must be stereotyped by an Activity (which is abstract) and its supplier end must be stereotyped by a Capability. But as it is not possible to show this constraint graphically the diagram does not communicate the needed information. We then use the "metaconstraint" dependency to visualize the constraint.

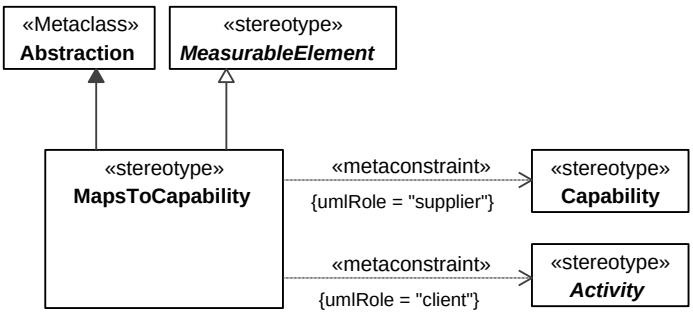

Figure 1 – MapsToCapability Stereotype

With the metaconstraint dependency added to the diagram (see Figure 1**)** which shows that MapsToCapability is a stereotype extending the Abstraction metaclass, that inherits the properties of a MeasurableElement and is used for modeling a relationship between an Activity (or its specializations) and a Capability (or its specializations). A Dependency stereotyped MapsToCapability must have its values for the client property stereotyped as an Activity, and its values for the supplier property must be stereotyped Capability.

Note – When stereotype extends Connector, the stereotype property umlRole has values "end[0].role" and "end[1].role." For example:

This is done because Connector has no direct "linkage" to the connected element; it links to the Connector Ends, which references the linked element. So, end[n] gives the reference to the ConnectorEnd, and role gives the reference to the linked element.

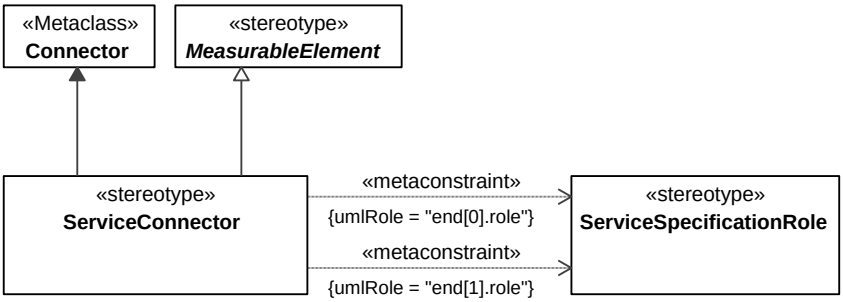

Figure 2 – Connector Extension

6.5.2 Metarelationship dependency

«metarelationship» is a stereotype for dependency, showing that certain domain concepts will be implemented using regular UML relationships.

For example: A Capability may depend on other Capabilities or be subtype of a Capability, but this concept cannot be visualized on the diagram:

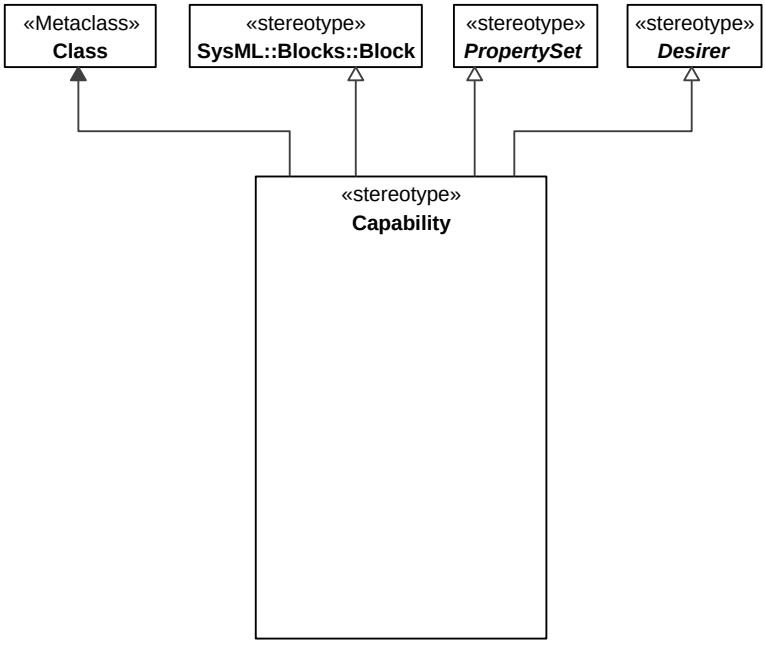

Figure 3 – Capabilities Generalization

We are using the «metarelationship» dependency to visualize the dependency and the generalization concept.

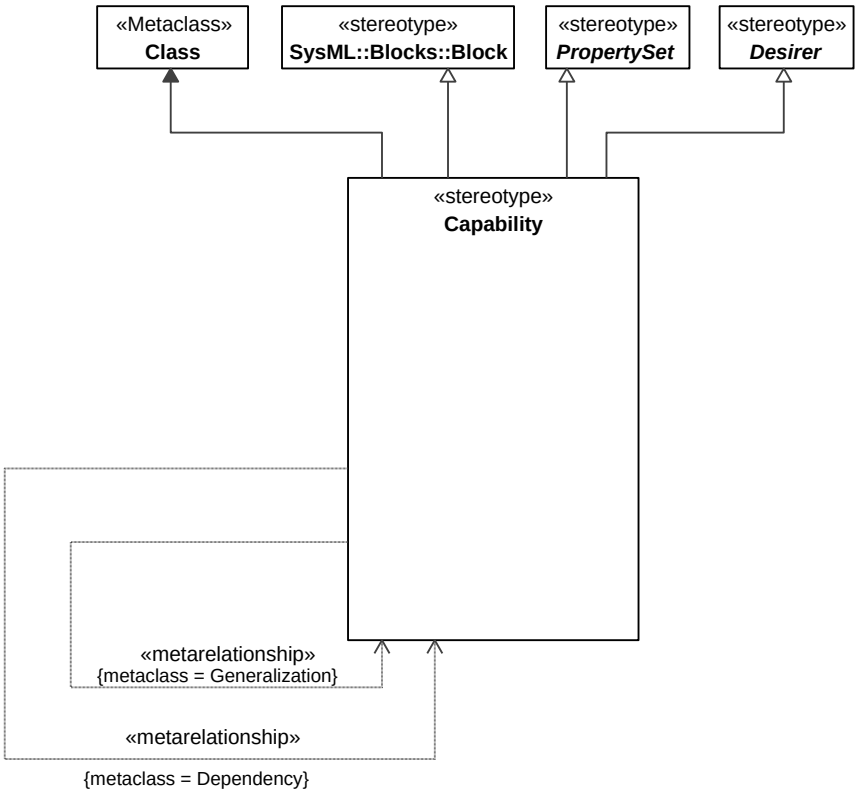

Figure 4 – Visualizing «metarelationship»

This diagram should be read as follows:

Capability may have other Capabilities related to it, using the UML Dependency metaclass and it may have sub types of Capabilities related to it, using the the UML Generalization metaclass.

The «metarelationship» dependency will appear only in the specification diagrams, but not the profile XMI.

6.5.3 Stereotyped relationship dependency

Although the «metarelationship» dependency creates a good way to show the constrained ends of the stereotyped relationship, it also creates some overhead when showing the relationship between two stereotypes.
For example, Figure 5 below shows that elements of subtype Achiever have a stereotyped relationship called AchievedEffect with elements of type ActualState.

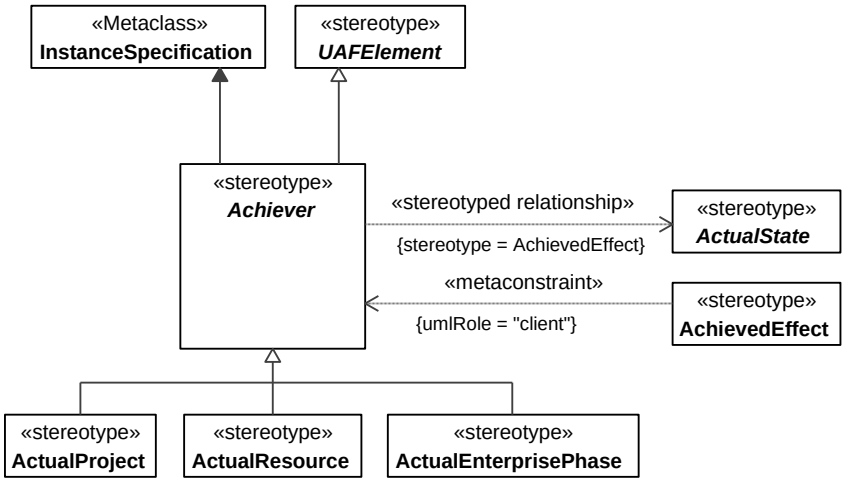

Figure 5 – Use of the AchievedEffect «stereotyped relationship» dependency

7.Part II – UAF Profile

UAFP imports the entire SysML profile and contains a set of constraints that specify which SysML stereotypes are applied to the UAFP elements. This is intended to provide more seamless integration with system modeling using SysML and to be able to fully leverage the capabilities of SysML in UAFP. An example of this is the integration of Requirements into the UAFP and also the use of Parametric Diagrams and integration of elements based upon instance specifications to allow the assessment of measures within an architecture developed using UAFP.

7.1 UAF

UAF is the top level profile root.

7.1.1 UAF::Dictionary

Stakeholders: Architects, users of the architecture, Capability Owners, Systems Engineers, Solution Providers.
Concerns: Definitions for all the elements in the architecture, libraries of environments and measurements.
Definition: Presents all the elements used in an architecture. Can be used specifically to capture:
a. elements and relationships that are involved in defining the environments applicable to capability, operational concept or set of systems.
b. measurable properties that can be used to support analysis such as KPIs, MoEs, TPIs etc.

Alias

Package: Dictionary

isAbstract: No

Generalization: MeasurableElement

Extension: Comment
Description

A metamodel Artifact used to define an alternative name for an element.

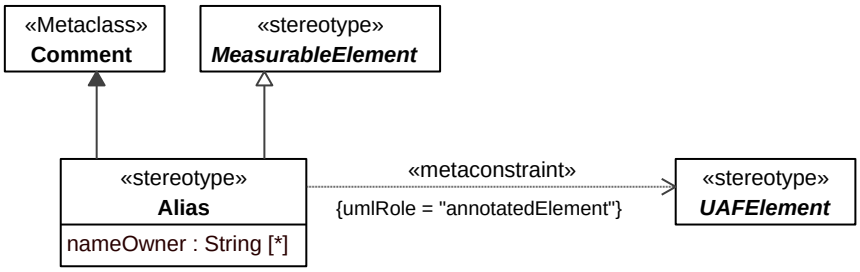

Figure 6 - Alias

Attributes

nameOwner : String[*] Someone or something that uses this alternative name.

Constraints

[1] Alias.annotatedElement Value for the annotatedElement metaproperty must be stereotyped by the specialization of «UAFElement».

Definition

Package: Dictionary

isAbstract: No

Generalization: MeasurableElement

Extension: Comment

Description

A comment containing a description of an element in the architecture.

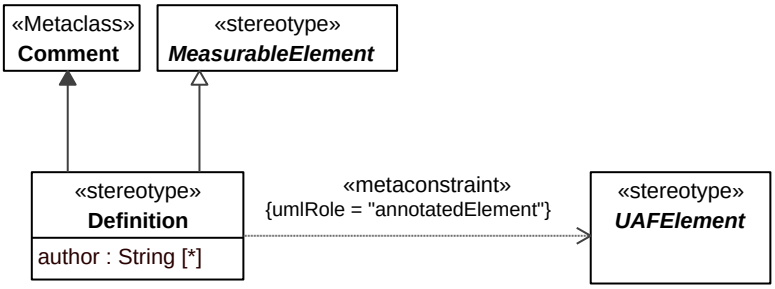

Figure 7 – Definition

Attributes

author : String[*] The original or current person (architect) responsible for the Definition.

Constraints

[1] Definition.annotatedElement Value for the annotatedElement metaproperty must be stereotyped by the specialization of «UAFElement».

SameAs

Package: Dictionary

isAbstract: No

Generalization: MeasurableElement

Extension: Dependency

Description

A dependency relationship that asserts that two elements refer to the same real-world thing.

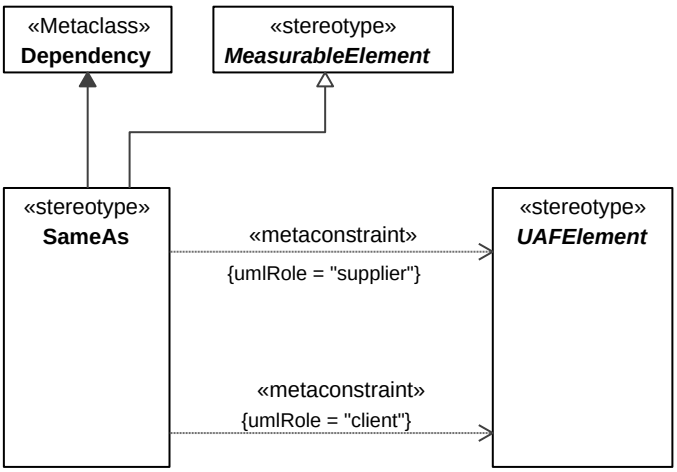

Figure 8 – SameAs

Constraints

[1] SameAs.client Values for the client metaproperty must be stereotyped by the specialization of «UAFElement».

[2] SameAs.supplier Values for the supplier metaproperty must be stereotyped by the specialization of «UAFElement».

7.1.2 UAF::Parameters

ActualCondition

Package: Parameters

isAbstract: No

Generalization: ActualPropertySet

Extension: InstanceSpecification

Description

The actual state of an environment or location describing its situation.

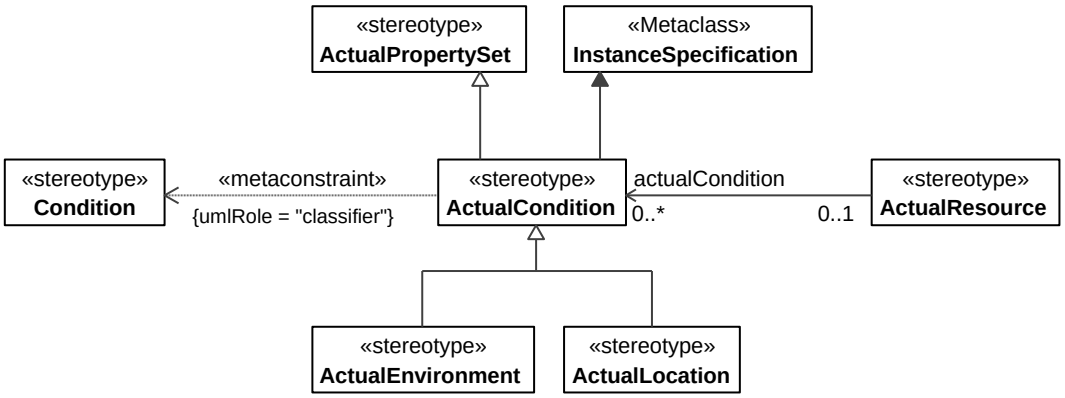

Figure 9 – ActualCondition

Constraints

[1] ActualCondition.classifier	Value for the classifier metaproperty has to be stereotyped «Condition» or its specializations.

ActualEnvironment

Package: Parameters

isAbstract: No

Generalization: ActualCondition

Extension: InstanceSpecification

Description

The ActualState that describes the circumstances of an Environment.

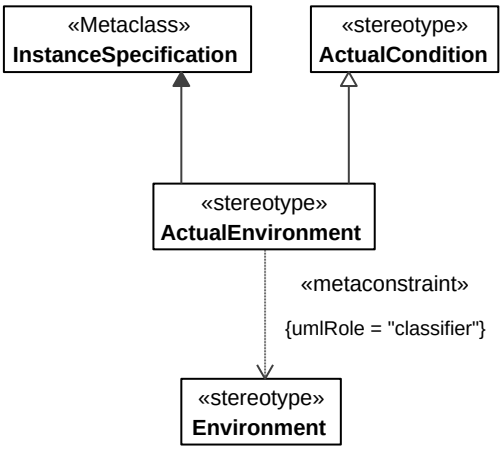

Figure 10 – ActualEnvironment

Constraints

[1] ActualEnvironment.classifier	Value for the classifier metaproperty has to be stereotyped «Environment» or its specializations.

ActualLocation

Package: Parameters

isAbstract: No

Generalization: ActualCondition

Extension: InstanceSpecification

Description

An ActualState that describes a physical location, for example using text to provide an address, Geo-coordinates, etc.

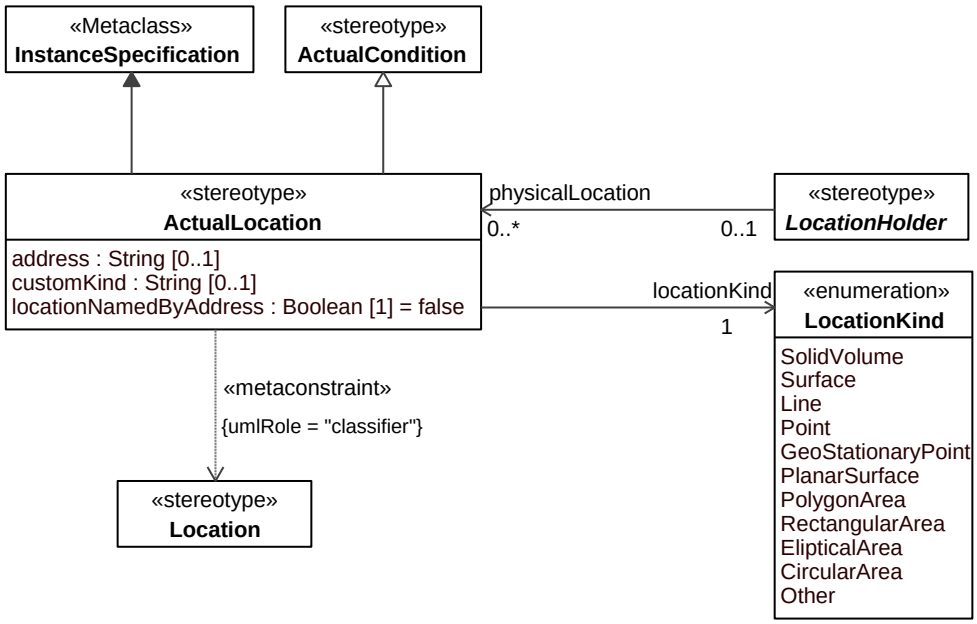

Figure 11 – ActualLocation

Attributes

address : String[0..1]	String describing the address of the ActualLocation, i.e. "1600 Pennsylvania avenue", "The White House"
customKind : String[0..1]	String describing a location kind that is not in the LocationKind enumerated list
locationNamedByAddress : Boolean[1]	Boolean that indicates if the ActualLocation address is embedded in the ActualLocation name. By default = false.

Associations

locationKind : LocationKind[1] Enumerated value describing the kind of ActualLocation.

Constraints

[1] ActualLocation.classifier Classifier metaproperty value must be stereotyped «Location» or its specializations.

ActualMeasurement

Package: Parameters

isAbstract: No

Generalization: ActualState

Extension: Slot

Description

An actual value that is applied to a Measurement.

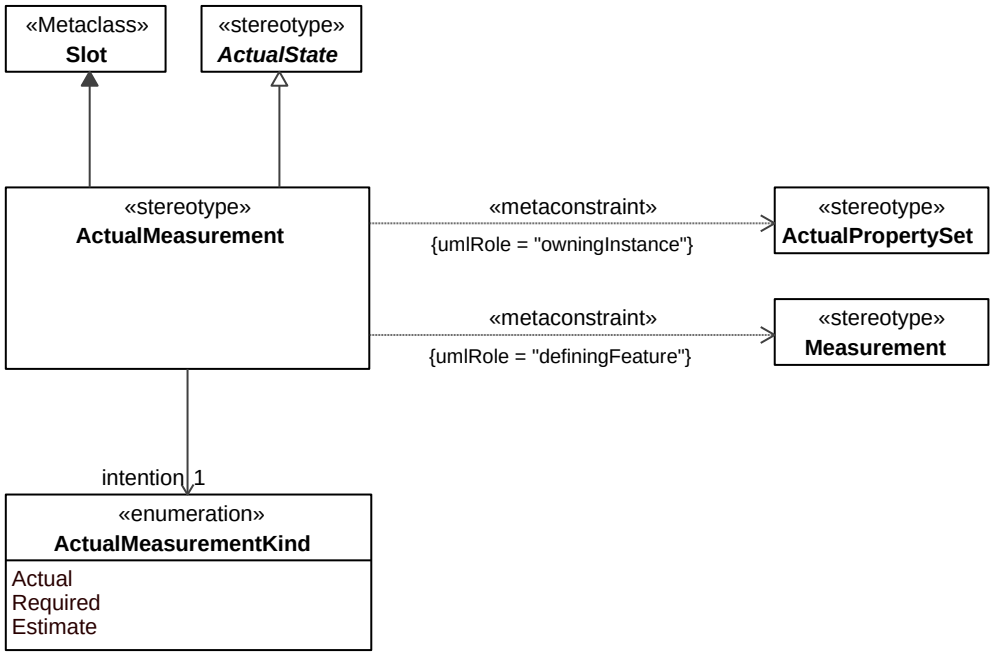

Figure 12 – ActualMeasurement

Associations

intention : ActualMeasurementKind[1] Enumerated value describing the intent of the ActualMeasurement.

Constraints

[1] ActualMeasurement.definingFeature Value for the definingFeature metaproperty must be stereotyped «Measurement» or its specializations.

[2] ActualMeasurement.owningInstance Value for the owningInstance metaproperty must be stereotyped «ActualPropertySet» or its specializations.

ActualMeasurementKind

Package: Parameters

isAbstract: No

Description

Enumeration of the possible kinds of ActualMeasurement. Its enumeration literals are:

- Actual - Indicates that the ActualMeasurement associated with the ActualMeasurementKind is a realworld value.
- Required - Indicates that the ActualMeasurement associated with the ActualMeasurementKind is a value that is expected to be achieved.
- Estimate - Indicates that the ActualMeasurement associated with the ActualMeasurementKind is an estimate of a realworld value.

ActualMeasurementSet

Package: Parameters

isAbstract: No

Generalization: ActualPropertySet

Extension: InstanceSpecification

Description

A set of ActualMeasurements.

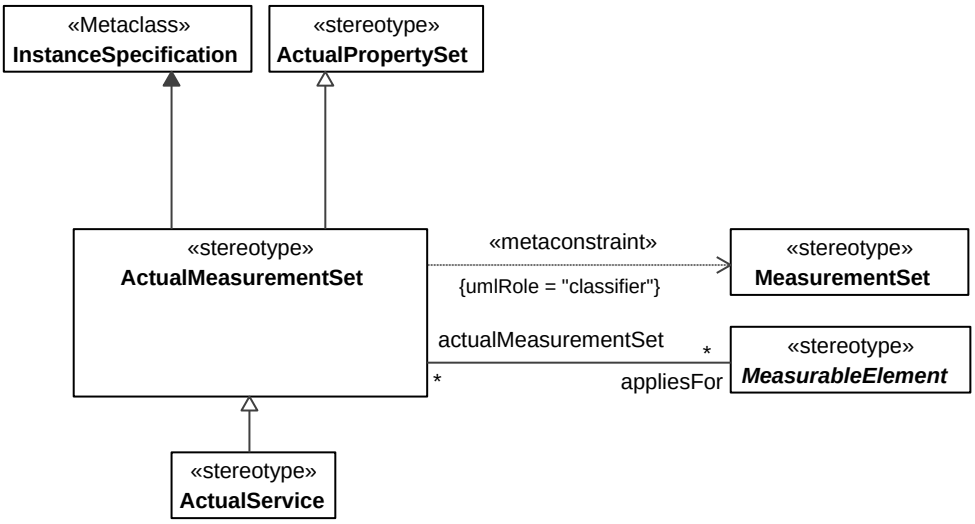

Figure 13 – ActualMeasurementSet

Associations

appliesFor : MeasurableElement[*] Relates the ActualMeasurementSet to the elements that are being measured.

Constraints

[1] ActualMeasurementSet.classifier Classifier metaproperty value must be stereotyped «MeasurementSet» or its specializations.

[2] ActualMeasurementSet.slot Value for the slot metaproperty must be stereotyped «ActualMeasurement» or its specializations.

ActualPropertySet

Package: Parameters

isAbstract: No

Generalization: ActualState

Extension: InstanceSpecification

Description

A set or collection of Actual properties.

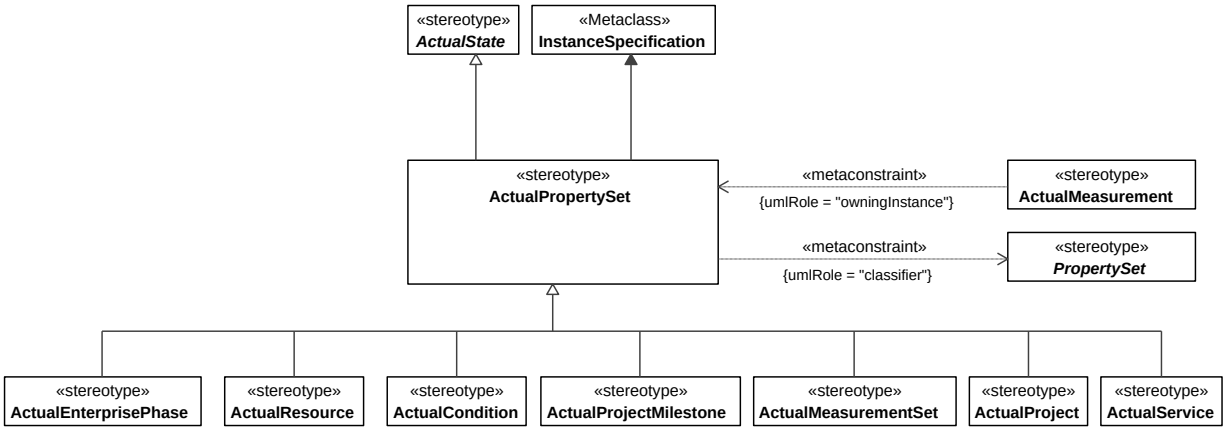

Figure 14 – ActualPropertySet

Constraints

[1] ActualPropertySet.classifier Value for the classifier metaproperty must be stereotyped by the specialization of «PropertySet».

Condition

Package: Parameters

isAbstract: No

Generalization: PropertySet, ValueType

Extension: DataType
Description

Defines the Location, Environment and/or GeoPoliticalExtent under which an OperationalActivity, Function or ServiceFunction can be performed.

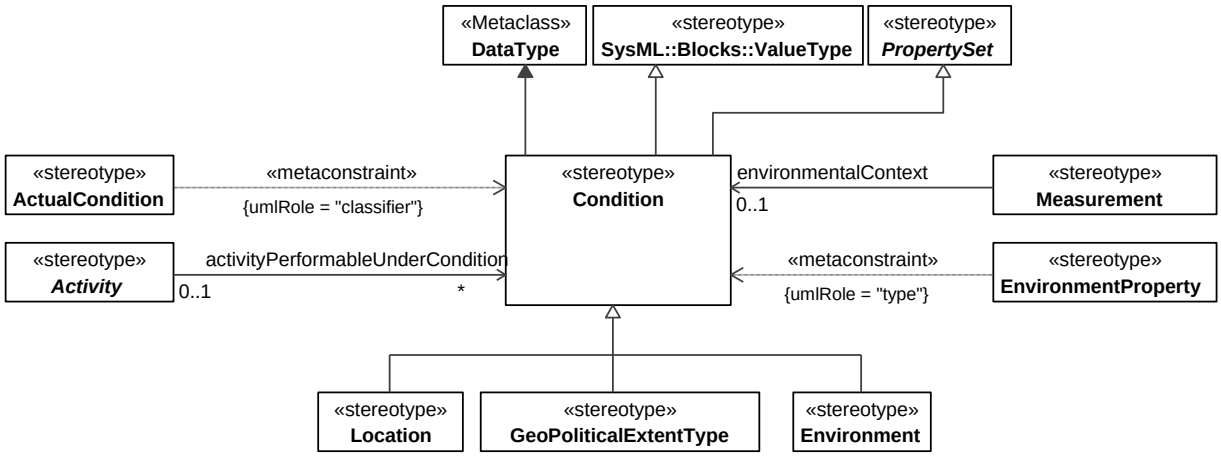

Figure 15 – Condition

Environment

Package: Parameters

isAbstract: No

Generalization: Condition

Extension: DataType
Description

A definition of the environmental factors in which something exists or functions. The definition of an Environment element can be further defined using EnvironmentKind.

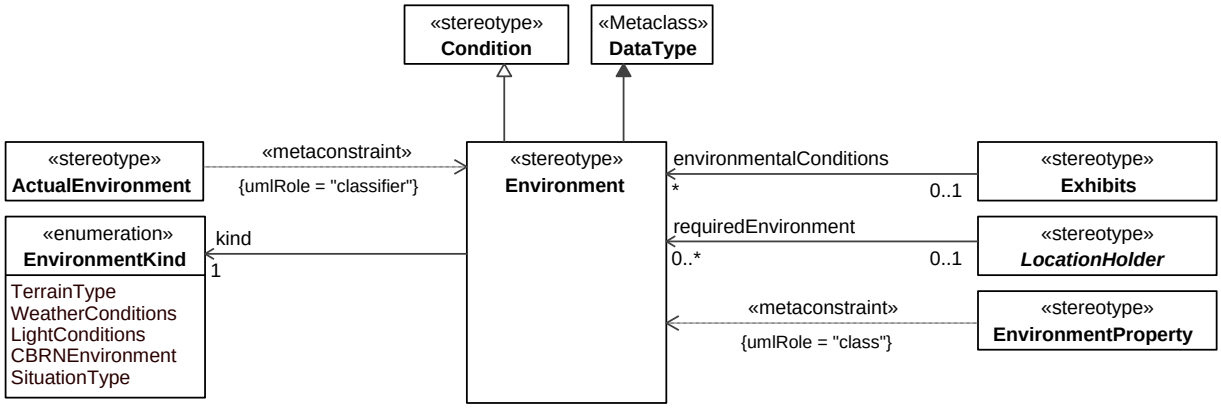

Figure 16 – Environment

Associations

kind : EnvironmentKind[1] Captures the kind of Environment.

EnvironmentKind

Package: Parameters

isAbstract: No
Description

Enumeration of the possible kinds of Environment. Its enumeration literals are:

- TerrainType - Indicates that the Environment associated with EnvironmentKind captures a kind of terrain used to describe the terrain state of an environment at a particular time (e.g. muddy, frozen ground, deep snow, etc.).
- WeatherConditions - Indicates that the Environment associated with EnvironmentKind captures a kind of weather condition (e.g. Typhoon, Hurricane, Very Hot, Humid etc.).
- LightConditions - Indicates that the Environment associated with EnvironmentKind captures a kind of light condition (e.g. broad daylight, dusk, moonlit, etc.).
- CBRNEnvironment - Indicates that the Environment associated with EnvironmentKind is of a Chemical, Biological, Radiological or Nuclear (CBRN) kind.
- SituationType - Indicates that the Environment associated with EnvironmentKind captures a kind of situation used to describe the types and levels of threat (e.g. Corrosive, Fire, Smoke, Peaceful etc.).

EnvironmentProperty

Package: Parameters

isAbstract: No

Generalization: MeasurableElement

Extension: Property
Description

A property of an Environment that is typed by a Condition. The kinds of Condition that can be represented are Location, GeoPoliticalExtentType and Environment.

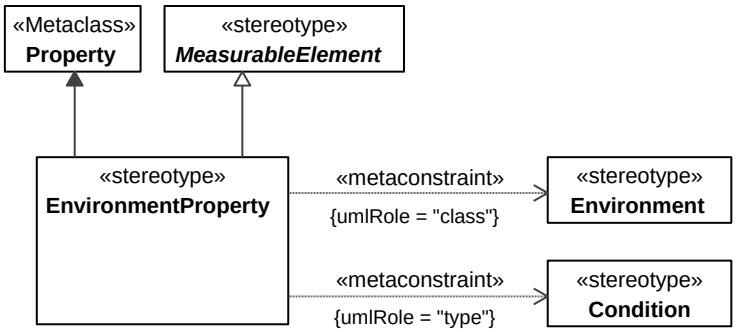

Figure 17 – EnvironmentProperty

Constraints

[1] EnvironmentalProperty.class	Value for the class metaproperty must be stereotyped «Environment» or its specializations.
[2] EnvironmentalProperty.type	Value for the type property must be stereotyped «Condition» or its specializations.

GeoPoliticalExtentType

Package: Parameters

isAbstract: No

Generalization: ResourceExchangeItem, OperationalExchangeItem, Condition

Extension: DataType
Description

A geospatial extent whose boundaries are defined by declaration or agreement by political parties.

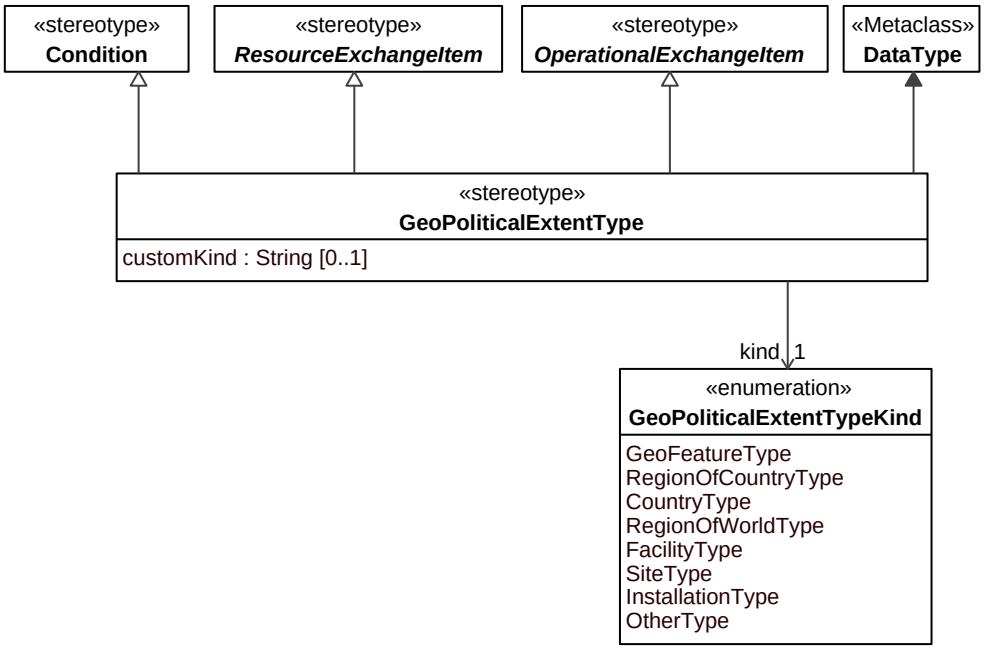

Figure 18 – GeoPoliticalExtentType

Attributes

customKind : String[0..1] Captures the kind of GeopoliticalExtentType if the GeoPoliticalExtentTypeKind has been set to "OtherType".

Associations

kind : GeoPoliticalExtentTypeKind[1] Captures the kind of GeopoliticalExtentType.

GeoPoliticalExtentTypeKind

Package: Parameters

isAbstract: No

Description

Enumeration of the possible kinds of GeoPoliticalExtentType. Its enumeration literals are:

- GeoFeatureType - Indicates that the GeoPoliticalExtentType associated with the GeoPoliticalExtentTypeKind is a type of object that encompasses meteorological, geographic, and control features mission significance.
- RegionOfCountryType - Indicates that the GeoPoliticalExtentType associated with the GeoPoliticalExtentTypeKind is a type of large, usually continuous segment of a political state, nation or its territory.
- CountryType - Indicates that the GeoPoliticalExtentType associated with the GeoPoliticalExtentTypeKind is a type of political state, nation or its territory.
- RegionOfWorldType - Indicates that the GeoPoliticalExtentType associated with the GeoPoliticalExtentTypeKind is a type of large, usually continuous segment of a surface or space; area.
- FacilityType - Indicates that the GeoPoliticalExtentType associated with the GeoPoliticalExtentTypeKind is a type of a real property entity consisting of underlying land and one or more of the following: a building, a structure (including linear structures), a utility system, or pavement.
- SiteType - Indicates that the GeoPoliticalExtentType associated with the GeoPoliticalExtentTypeKind is a type of Physical (geographic) location that is or was owned by, leased to, or otherwise possessed. Each site is assigned to a single installation. A site may exist in one of three forms: (1) Land only, where there are no facilities present and where the land consists of either a single land parcel or two or more contiguous land parcels. (2) Facility or facilities only, where the underlying land is neither owned nor controlled by the government. A stand-alone facility can be a site. If a facility is not a stand-alone facility, it must be

assigned to a site. (3). Land and all the facilities thereon, where the land consists of either a single land parcel or two or more contiguous land parcels.

- InstallationType - Indicates that the GeoPoliticalExtentType associated with the GeoPoliticalExtentTypeKind is a type of base, camp, post, station, yard, center, or other activity, including leased facilities, without regard to the duration of operational control. An installation may include one or more sites.
- OtherType - Indicates that the GeoPoliticalExtentType associated with the GeoPoliticalExtentTypeKind is a type not covered by the standard GeoPoliticalExtentTypeKinds.

Location

Package: Parameters

isAbstract: No

Generalization: ConceptItem, Condition

Extension: DataType
Description

A specification of the generic area in which a LocationHolder is required to be located.

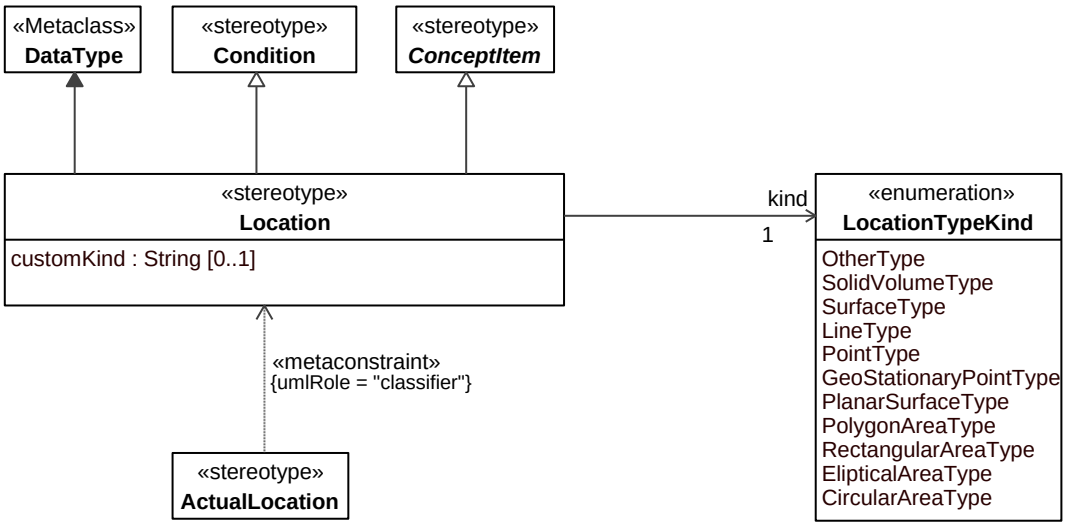

Figure 19 – Location

Attributes

customKind : String[0..1] Captures the kind of Location if the LocationTypeKind has been set to "OtherType".
Associations

kind : LocationTypeKind[1] Captures the kind of Location.

LocationHolder

Package: Parameters

isAbstract: Yes

Generalization: UAFElement

Extension: Element
Description

Abstract grouping used to define elements that are allowed to be associated with a Location.

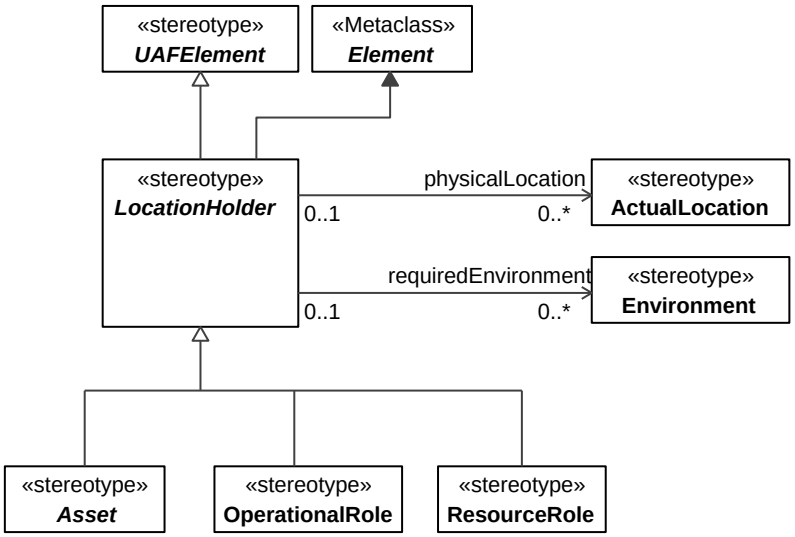

Figure 20 – LocationHolder

Associations

physicalLocation : ActualLocation[0..*]	Relates a LocationHolder (i.e. OperationalPerformer, OperationalRole, ResourceRole etc.) to its ActualLocation.
requiredEnvironment : Environment[0..*]	Relates a LocationHolder (i.e. OperationalPerformer, OperationalRole, ResourceRole etc.) to the Environment in which it is required to perform/be used.

LocationKind

Package: Parameters

isAbstract: No

Description

Enumeration of the possible kinds of location applicable to an ActualLocation. Its enumeration literals are:

- SolidVolume - Indicates that the ActualLocation associated with the LocationKind is the amount of space occupied by a three-dimensional object of definite shape; not liquid or gaseous.
- Surface - Indicates that the ActualLocation associated with the LocationKind is a portion of space having length and breadth but no thickness or regards to time.
- Line - Indicates that the ActualLocation associated with the LocationKind is a geometric figure formed by a point moving along a fixed direction and the reverse direction.
- Point - Indicates that the ActualLocation associated with the LocationKind is a unidimensional Individual.
- GeoStationaryPoint - Indicates that the ActualLocation associated with the LocationKind is a unidimensional Individual.
- PlanarSurface - Indicates that the ActualLocation associated with the LocationKind is a two-dimensional portion of space.
- PolygonArea - Indicates that the ActualLocation associated with the LocationKind is a space enclosed by a polygon.
- RectangularArea - Indicates that the ActualLocation associated with the LocationKind is a space enclosed by a rectangle.
- ElipticalArea - Indicates that the ActualLocation associated with the LocationKind is a space enclosed by an ellipse.
- CircularArea - Indicates that the ActualLocation associated with the LocationKind is a space enclosed by a circle.
- Other - Indicates that the ActualLocation associated with the LocationKind is a LocationKind that is not on the enumerated list.

LocationTypeKind

Package: Parameters

isAbstract: No
Description

Enumeration of the possible kinds of location type that are applicable to a Location. Its enumeration literals are:

- OtherType - Indicates that the Location associated with the LocationTypeKind describes a type of is a LocationKindType that is not on the enumerated list.
- SolidVolumeType - Indicates that the Location associated with the LocationTypeKind describes a type of amount of space occupied by a three-dimensional object of definite shape; not liquid or gaseous.
- SurfaceType - Indicates that the Location associated with the LocationTypeKind describes a type of portion of space having length and breadth but no thickness or regards to time.
- LineType - Indicates that the Location associated with the LocationTypeKind describes a type of geometric figure formed by a point moving along a fixed direction and the reverse direction.
- PointType - Indicates that the Location associated with the LocationTypeKind describes a type of unidimensional Individual.
- GeoStationaryPointType - Indicates that the Location associated with the LocationTypeKind describes a type of unidimensional Individual.
- PlanarSurfaceType - Indicates that the Location associated with the LocationTypeKind describes a type of is a two-dimensional portion of space.
- PolygonAreaType - Indicates that the Location associated with the LocationTypeKind describes a type of space enclosed by a polygon.
- RectangularAreaType - Indicates that the Location associated with the LocationTypeKind describes a type of space enclosed by a rectangle.
- ElipticalAreaType - Indicates that the Location associated with the LocationTypeKind describes a type of space enclosed by an ellipse.
- CircularAreaType - Indicates that the Location associated with the LocationTypeKind describes a type of space enclosed by a circle.

MeasurableElement

Package: Parameters

isAbstract: Yes

Generalization: UAFElement

Extension: Element
Description

Abstract grouping for elements that can be measured by applying MeasurementSets to them.

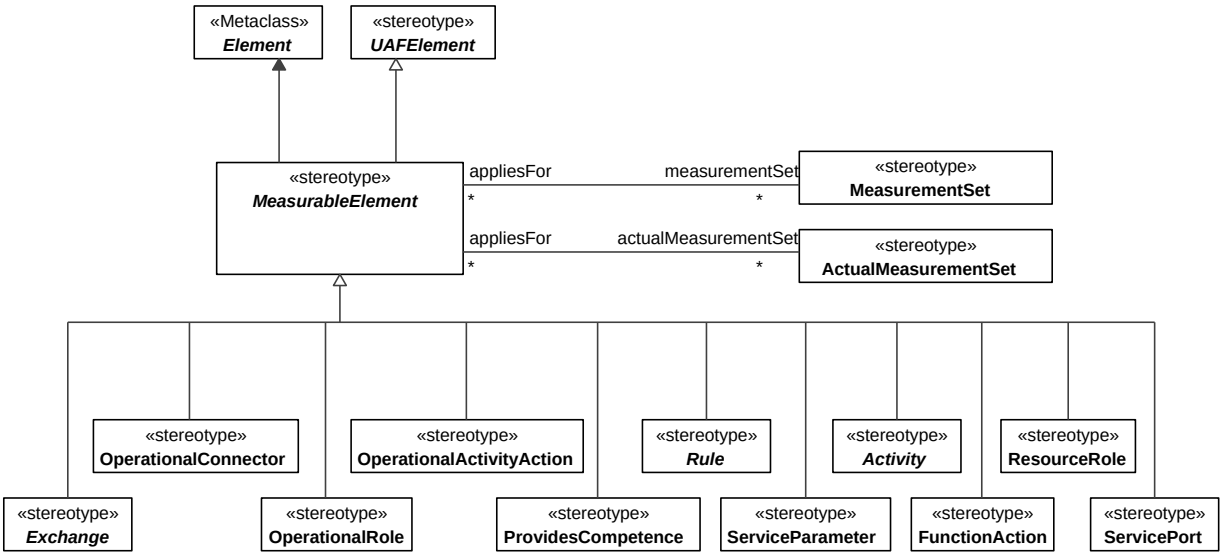

Figure 21 – MeasurableElement

Associations

actualMeasurementSet : ActualMeasurementSet[*]	Relates the MeasurableElement to the ActualMeasurementSet that provides its ActualMeasurements.
measurementSet : MeasurementSet[*]	Relates the MeasurableElement to the MeasurementSet that provides its Measurements by which it can be measured.

Measurement

Package: Parameters

isAbstract: No

Generalization: MeasurableElement

Extension: Property

Description

A property of an element representing something in the physical world, expressed in amounts of a unit of measure.

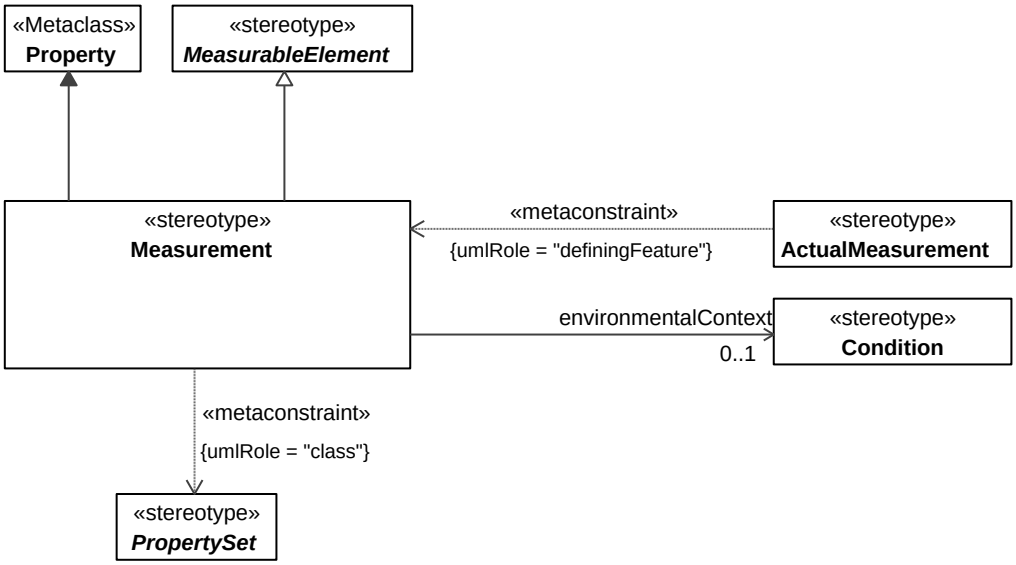

Figure 22 – Measurement

Associations

environmentalContext : Condition[0..1] Relates the Measurement to the Condition (which provides the environementalContext) under which the Measurement is expected to be taken.

Constraints

[1] Measurement.class Value for the class metaproperty must be stereotyped by the specialization of «PropertySet».

MeasurementSet

Package: Parameters

isAbstract: No

Generalization: PropertySet, ValueType

Extension: DataType

Description

A collection of Measurements.

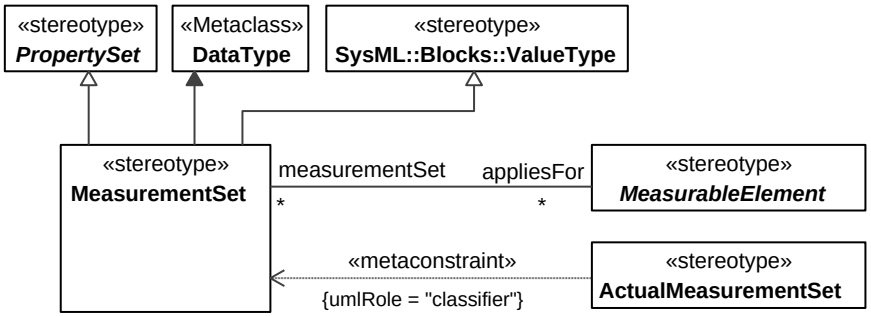

Figure 23 – MeasurementSet

Associations

appliesFor : MeasurableElement[*] Relates the MeasurementSet to the MeasurableElement that it is applicable to.

PropertySet

Package: Parameters

isAbstract: Yes

Generalization: UAFElement

Extension: Element

Description

An abstract grouping of architectural elements that can own Measurements.

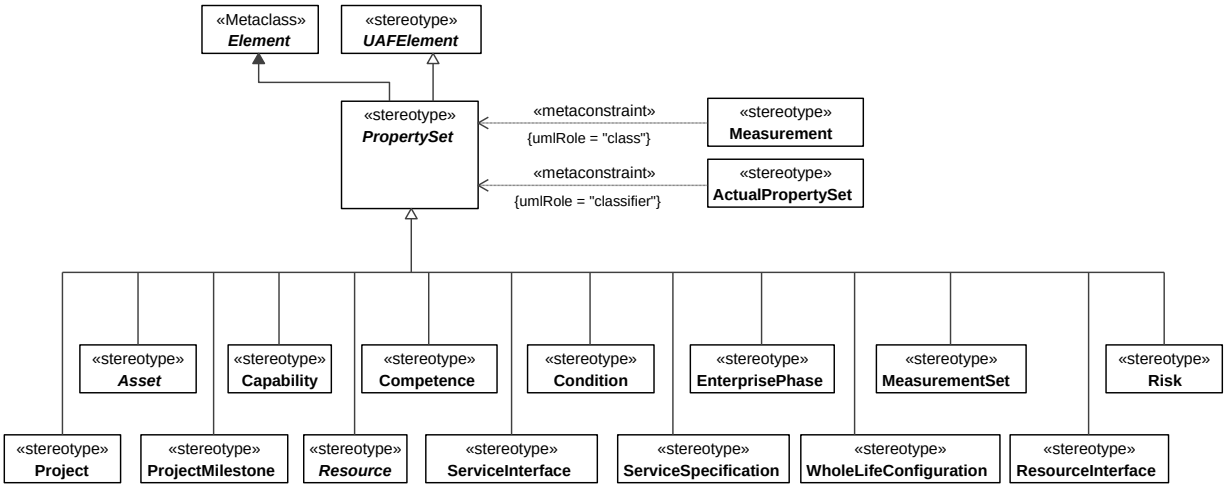

Figure 24 – PropertySet

7.1.3 UAF::Metadata

Stakeholders: Enterprise Architects, people who want to discover the architecture, Technical Managers.
Concerns: Captures meta-data relevant to the entire architecture
Definition: Provide information pertinent to the entire architecture. Present supporting information rather than architectural models.

7.1.3.1 UAF::Metadata::Taxonomy

Contains the elements that contribute to the Metadata Taxonomy Viewpoint.

ActualState

Package: Taxonomy

isAbstract: Yes

Generalization: UAFElement

Extension: Element
Description

Abstract element that applies temporal extent to a set of elements realized as Instance Specifications.

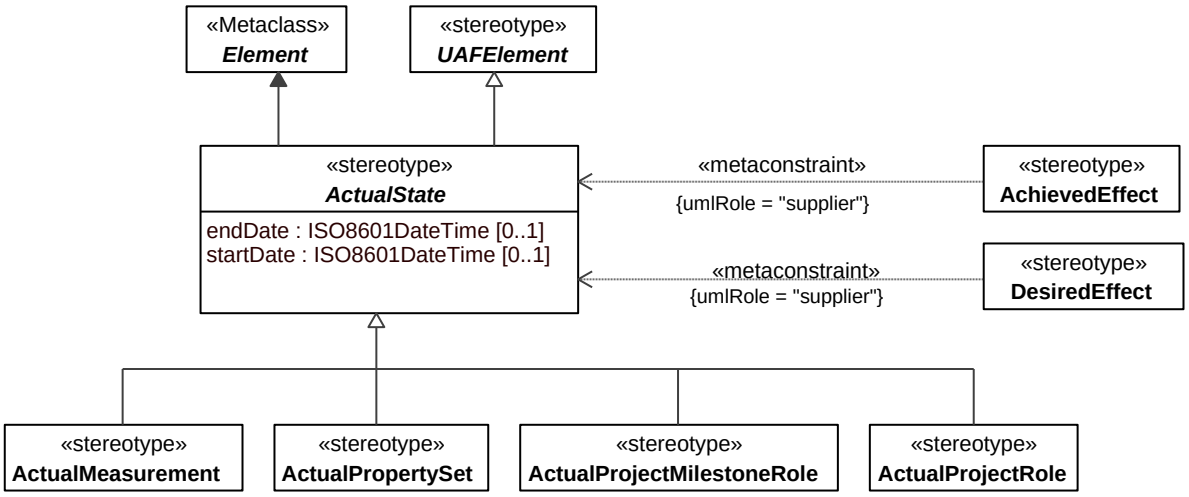

Figure 25 – ActualState

Attributes

endDate : ISO8601DateTime[0..1] End time for all "actual" elements.

startDate : ISO8601DateTime[0..1] Start time for all "actual" elements.

ISO8601DateTime

Package: Taxonomy

isAbstract: No

Generalization: UAFElement

Extension: LiteralString

Description

A date and time specified in the ISO8601 date-time format including timezone designator (TZD): YYYY-MM-DDThh:mm:ssTZD.

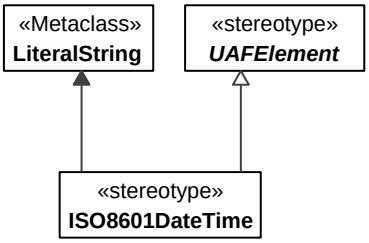

Figure 26 – ISO8601DateTime

7.1.3.2 UAF::Metadata::Connectivity

Contains the elements that contribute to the Metadata Connectivity Viewpoint.

Exchange

Package: Connectivity

isAbstract: Yes

Generalization: MeasurableElement, ItemFlow

Extension: InformationFlow

Description

Abstract grouping for OperationalExchanges and ResourceExchanges that exchange Resources.

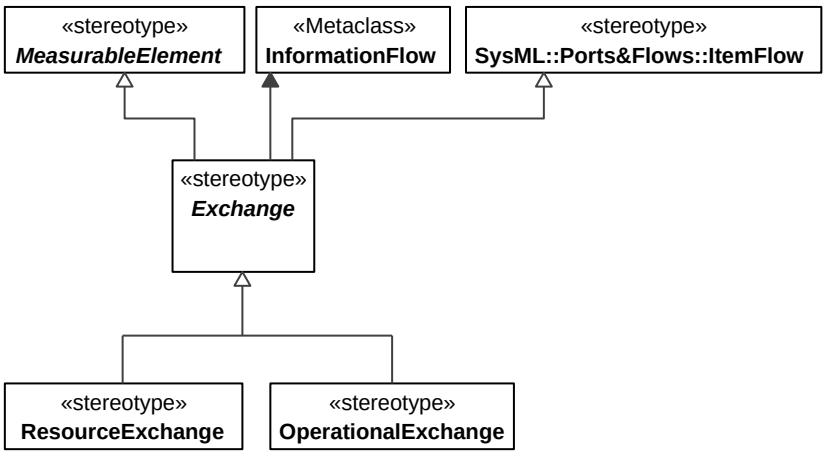

Figure 27 – Exchange

Resource

Package: Connectivity

isAbstract: Yes

Generalization: PropertySet

Extension: Element
Description

Abstract element grouping for all elements that can be conveyed by an Exchange.

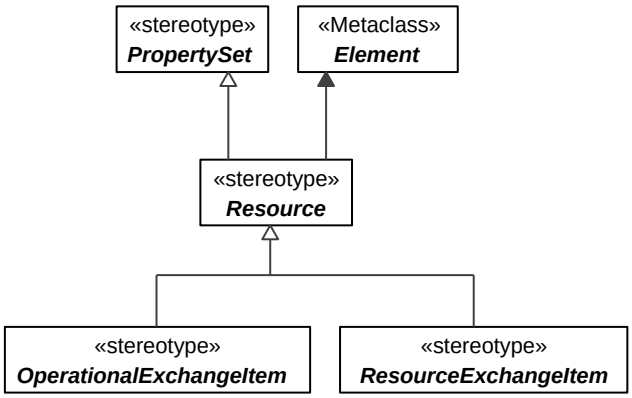

Figure 28 – Resource

7.1.3.3 UAF::Metadata::Processes

Contains the elements that contribute to the Metadata Processes Viewpoint.

Activity

Package: Processes

isAbstract: Yes

Generalization: MeasurableElement

Extension: Activity
Description

An abstract element that represents a behavior or process (i.e. a Function or OperationalActivity) that can be performed by a Performer.

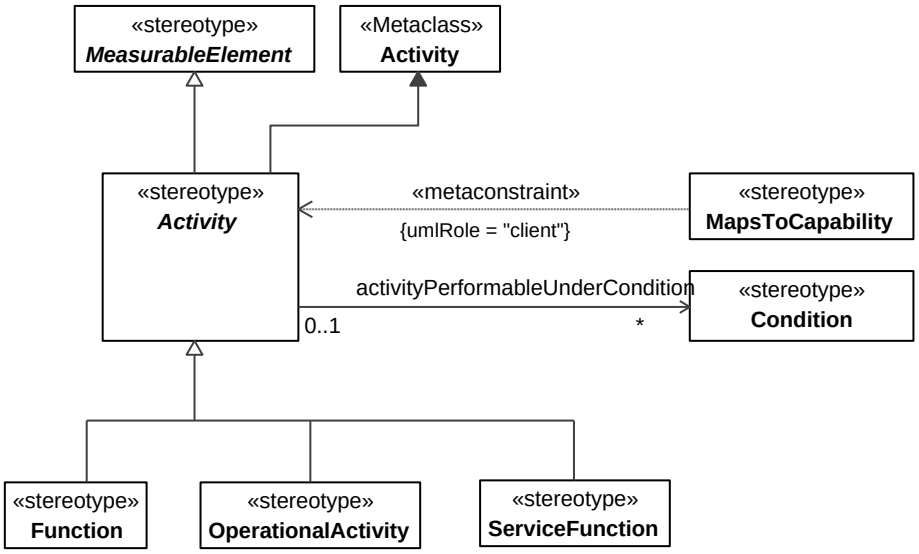

Figure 29 – Activity

Associations

activityPerformableUnderCondition : Condition[*] The environment under which an activity is performed.

CapableElement

Package: Processes

isAbstract: Yes

Generalization: UAFElement

Extension: Element

Description

An abstract element that represents a structural element that can perform behaviors (i.e. OperationalActivity).

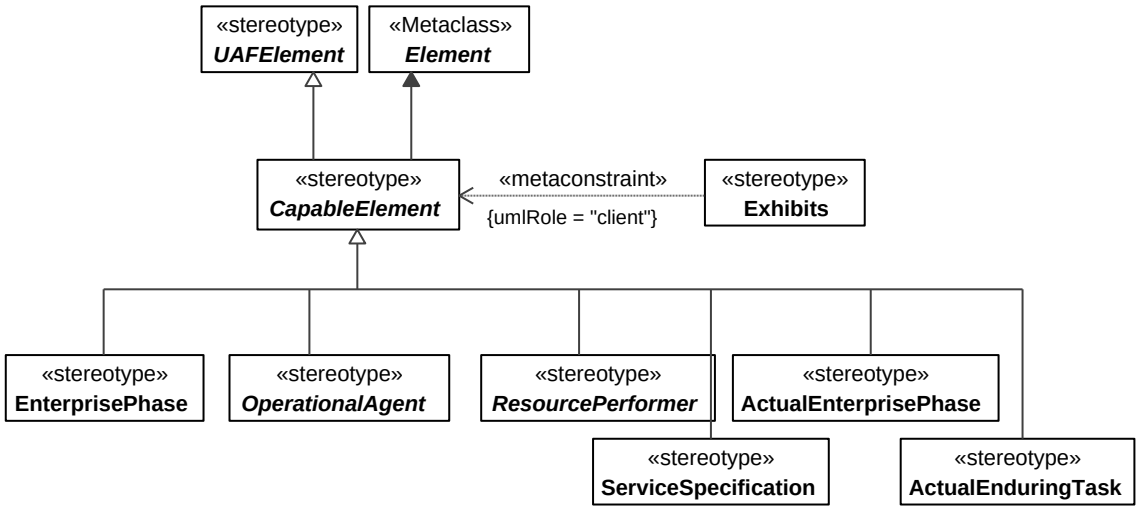

Figure 30 – CapableElement

IsCapableToPerform

Package: Processes

isAbstract: No

Generalization: MeasurableElement, Allocate

Extension: Abstraction

Description

An Abstraction relationship defining the traceability between the CapableElements to the Activities that they can perform.

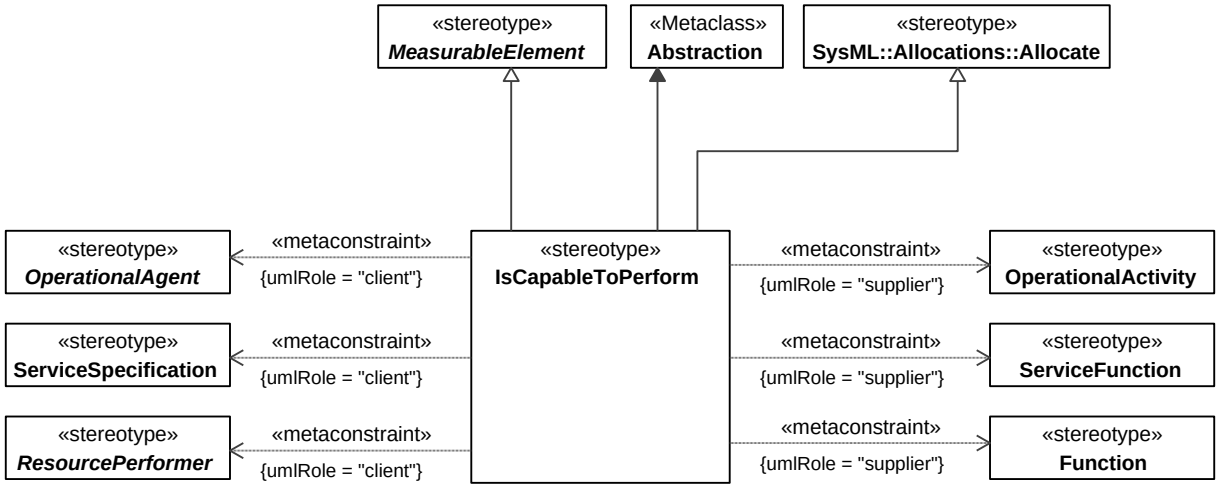

Figure 31 – IsCapableToPerform

Constraints

[1] IsCapableToPerform.client	In case of value for IsCapableToPerform.supplier is stereotyped: a. «OperationalActivity» or its specializations, values for the client metaproperty must be stereotyped by any of specializations of «OperationalAgent», b. «ServiceFunction» or its specializations, values for the client metaproperty must be stereotyped «ServiceSpecification» or its specializations, c. «Function» or its specializations, values for the client metaproperty must be stereotyped by any of specializations of «ResourcePerformer».
[2] IsCapableToPerform.supplier	In case of value for IsCapableToPerform.client is stereotyped: a. by a specialization of «OperationalAgent», values for the supplier metaproperty must be stereotyped «OperationalActivity» or its specializations, b. «ServiceSpecification» or its specializations, values for the supplier metaproperty must be stereotyped «ServiceFunction» or its specializations, c. by a specialization of «ResourcePerformer», values for the supplier metaproperty must be stereotyped «Function» or its specializations.

PerformsInContext

Package: Processes

isAbstract: No

Generalization: MeasurableElement, Allocate

Extension: Abstraction

Description

An abstraction relationship that relates an OperationalAction to a OperationalRole, or a FunctionAction to a ResourceRole. It indicates that the action can be carried out by the role when used in a specific context or configuration.

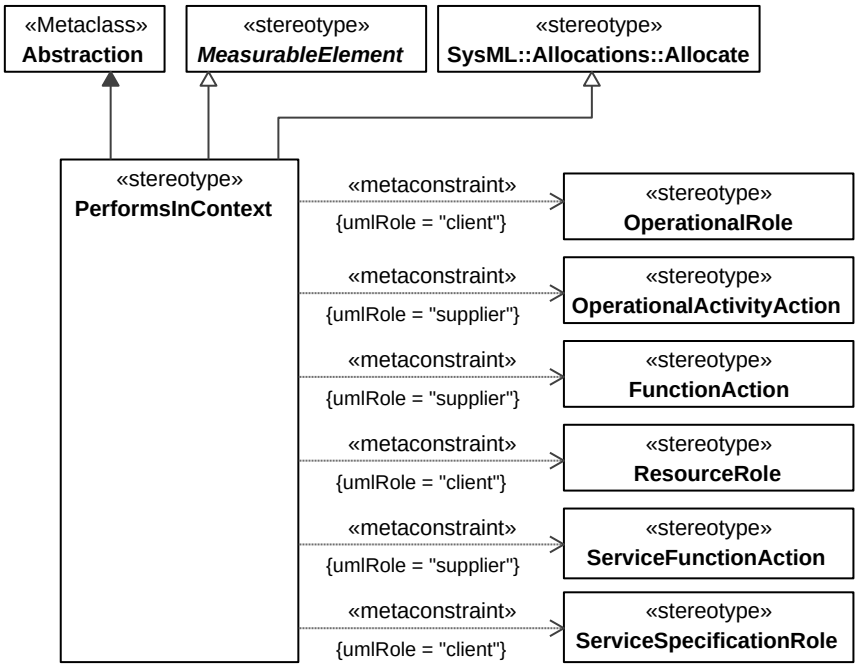

Figure 32 – PerformsInContext

Constraints

[1] PerformsInContext.client	In case of value for PerformsInContext.supplier is stereotyped: a. «OperationalActivityAction» or its specializations, values for the client metaproperty must be stereotyped «OperationalRole» or its specializations, b. «ServiceFunctionAction» or its specializations, values for the client metaproperty must be stereotyped «ServiceSpecificationRole» or its specializations, c. «FunctionAction» or its specializations, values for the client metaproperty must be stereotyped «ResourceRole» or its specializations.
[2] PerformsInContext.supplier	In case of value for PerformsInContext.client is stereotyped: a. «OperationalRole» or its specializations, values for the supplier metaproperty must be stereotyped «OperationalActivityAction» or its specializations, b. «ServiceSpecificationRole» or its specializations, values for the supplier metaproperty must be stereotyped «ServiceFunctionAction» or its specializations, c. «ResourceRole» or its specializations, values for the supplier metaproperty must be stereotyped «FunctionAction» or its specializations.

7.1.3.4 UAF::Metadata::Information

Contains the elements that contribute to the Metadata Information Viewpoint.

ArchitectureMetadata

Package: Information

isAbstract: No

Generalization: Metadata

Extension: Comment
Description

Information associated with an ArchitecturalDescription, that supplements the standard set of tags used to summarize the Architecture. It states things like what methodology was used, notation, etc.

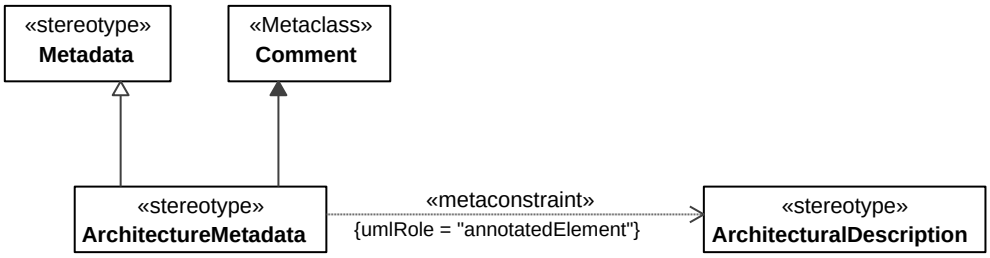

Figure 33 – ArchitectureMetadata

Constraints

[1] ArchitectureMetadata.annotatedElement — Value for the annotatedElement metaproperty must be stereotyped «ArchitecturalDescription» or its specializations.

Information

Package: Information

isAbstract: No

Generalization: MeasurableElement

Extension: Comment

Description

A comment that describes the state of an item of interest in any medium or form -- and is communicated or received.

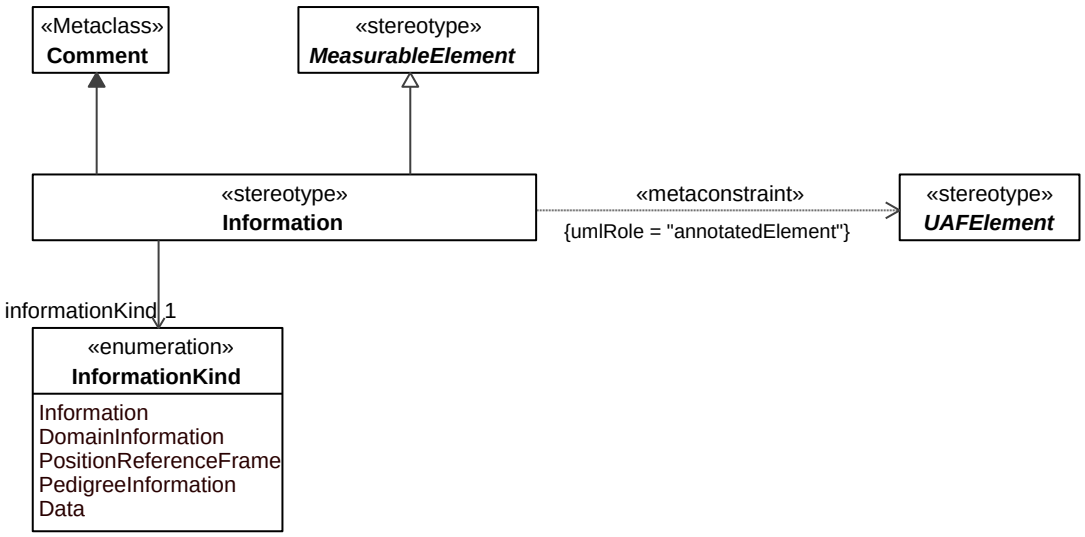

Figure 34 – Information

Associations

informationKind : InformationKind[1] — Captures the kind of information.

Constraints

[1] Information.annotatedElement — Value for the annotatedElement metaproperty must be stereotyped by a specialization of «ConceptItem».

InformationKind

Package: Information

isAbstract: No

Description

Enumeration of the possible kinds of Information. Its enumeration literals are:

- Information - Indicates that the Information associated with the InformationKind describes the state of a something of interest that is materialized -- in any medium or form -- and communicated or received.
- DomainInformation - Indicates that the Information associated with the InformationKind describes information within the scope or domain of the architecture.
- PositionReferenceFrame - Indicates that the Information associated with the InformationKind describes an arbitrary set of axes with reference to which the position or motion of something is described or physical laws are formulated.
- PedigreeInformation - Indicates that the Information associated with the InformationKind describes information pedigree.
- Data - Indicates that the Information associated with the InformationKind describes the representation of information in a formalized manner suitable for communication, interpretation, or processing by humans or by automatic means. Examples could be whole models, packages, entities, attributes, classes, domain values, enumeration values, records, tables, rows, columns, and fields.

Metadata

Package: Information

isAbstract: No

Generalization: MeasurableElement

Extension: Comment
Description

A comment that can be applied to any element in the architecture. The attributes associated with this element details the relationship between the element and its related dublinCoreElement, metaDataScheme, category and name. This allows the element to be referenced using the Semantic Web.

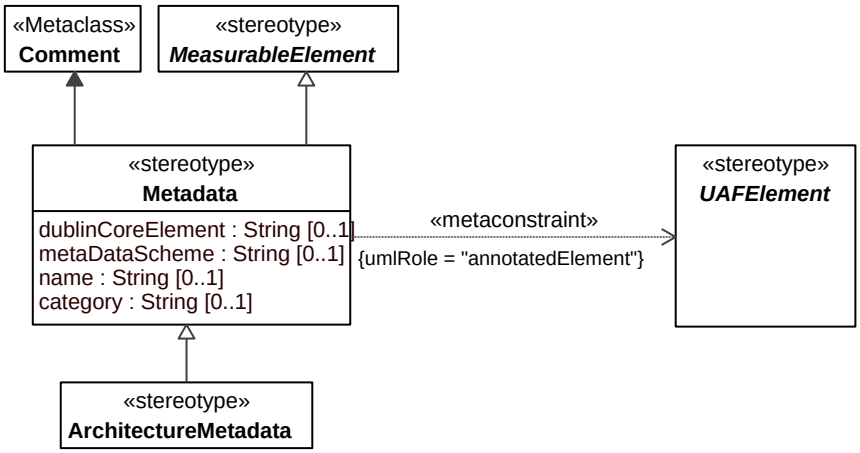

Figure 35 – Metadata

Attributes

category : String[0..1]	Defines the category of a Metadata element example: http://purl.org/dc/terms/abstract.
dublinCoreElement : String[0..1]	A metadata category that is a DublinCore tag.
metaDataScheme : String[0..1]	A representation scheme that defines a set of Metadata.
name : String[0..1]	The name of the Metadata.

Constraints

[1] Metadata.annotatedElement	Value for the annotatedElement metaproperty must be stereotyped by a specialization of «UAFElement».

7.1.3.5 UAF::Metadata::Constraints

Contains the elements that contribute to the Metadata Constraints Viewpoint.

Rule

Package: Constraints

isAbstract: Yes

Generalization: MeasurableElement

Extension: Constraint
Description

An abstract grouping for all types of constraint (i.e. an OperationalConstraint could detail the rules of accountancy best practice).

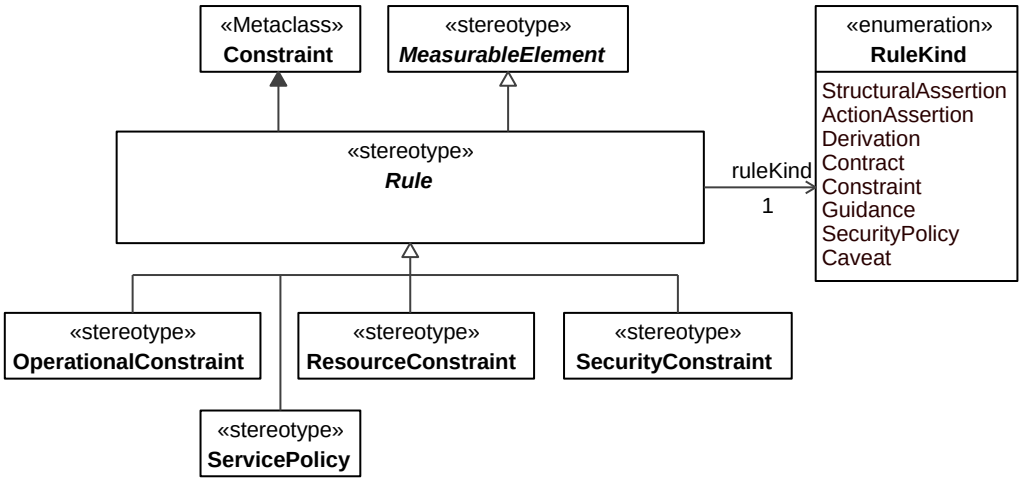

Figure 36 – Rule

Associations

ruleKind : RuleKind[1] Captures the kind of Rule that is being described.

RuleKind

Package: Constraints

isAbstract: No
Description

Enumeration of the possible kinds of Rules applicable to constraints. Its enumeration literals are:

- StructuralAssertion - Indicates that the Rule associated with the RuleKind is a statement that details that something of importance either exists as a concept of interest or exists in relationship to another thing of interest.
- ActionAssertion - Indicates that the Rule associated with the RuleKind is a statement that concerns some dynamic aspect.
- Derivation - Indicates that the Rule associated with the RuleKind is a statement that details a Rule derived from another Rule.
- Contract - Indicates that the Rule associated with the RuleKind is a statement that details a consent among parties regarding the terms and conditions of activities that said parties participate in.
- Constraint - Indicates that the Rule associated with the RuleKind is a statement that details a limitation, e.g. business rule, restraint, operational limitation.
- Guidance - Indicates that the Rule associated with the RuleKind is a statement that details an authoritative statement intended to lead or steer the execution of actions.
- SecurityPolicy - Indicates that the Rule associated with the RuleKind is a statement that details a constraint that specifies policy for information handling, physical security, encryption, etc.
- Caveat - Indicates that the Rule associated with the RuleKind is a statement that details alternate conditions under which the rule is not valid.

7.1.3.6 UAF::Metadata::Traceability

Contains the elements that contribute to the Metadata Traceability Viewpoint.

ArchitecturalReference

Package: Traceability

isAbstract: No

Generalization: MeasurableElement

Extension: Dependency
Description

A dependency relationship that specifies that one architectural description refers to another.

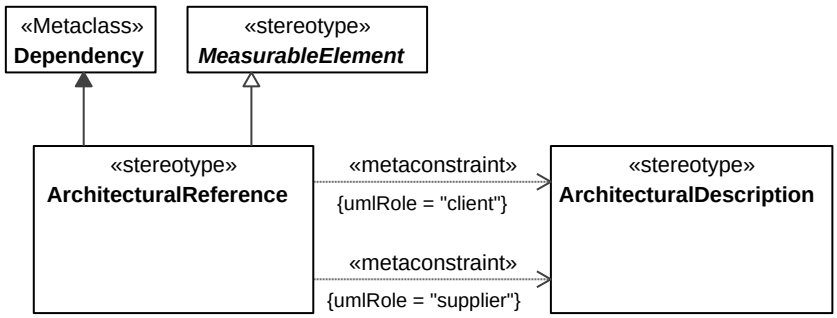

Figure 37 – ArchitecturalReference

Constraints

[1] ArchitecturalReference.client	Value for the client metaproperty must be stereotyped «ArchitecturalDescription» or its specializations.
[2] ArchitecturalReference.supplier	Value for the supplier metaproperty must be stereotyped «ArchitecturalDescription» or its specializations.

Implements

Package: Traceability

isAbstract: No

Generalization: MeasurableElement, Allocate

Extension: Abstraction
Description

An Abstraction relationship that defines how an element in the upper layer of abstraction is implemented by a semantically equivalent element (i.e. tracing the OperationalActivities to the Functions that implement them) in the lower level of abstraction.

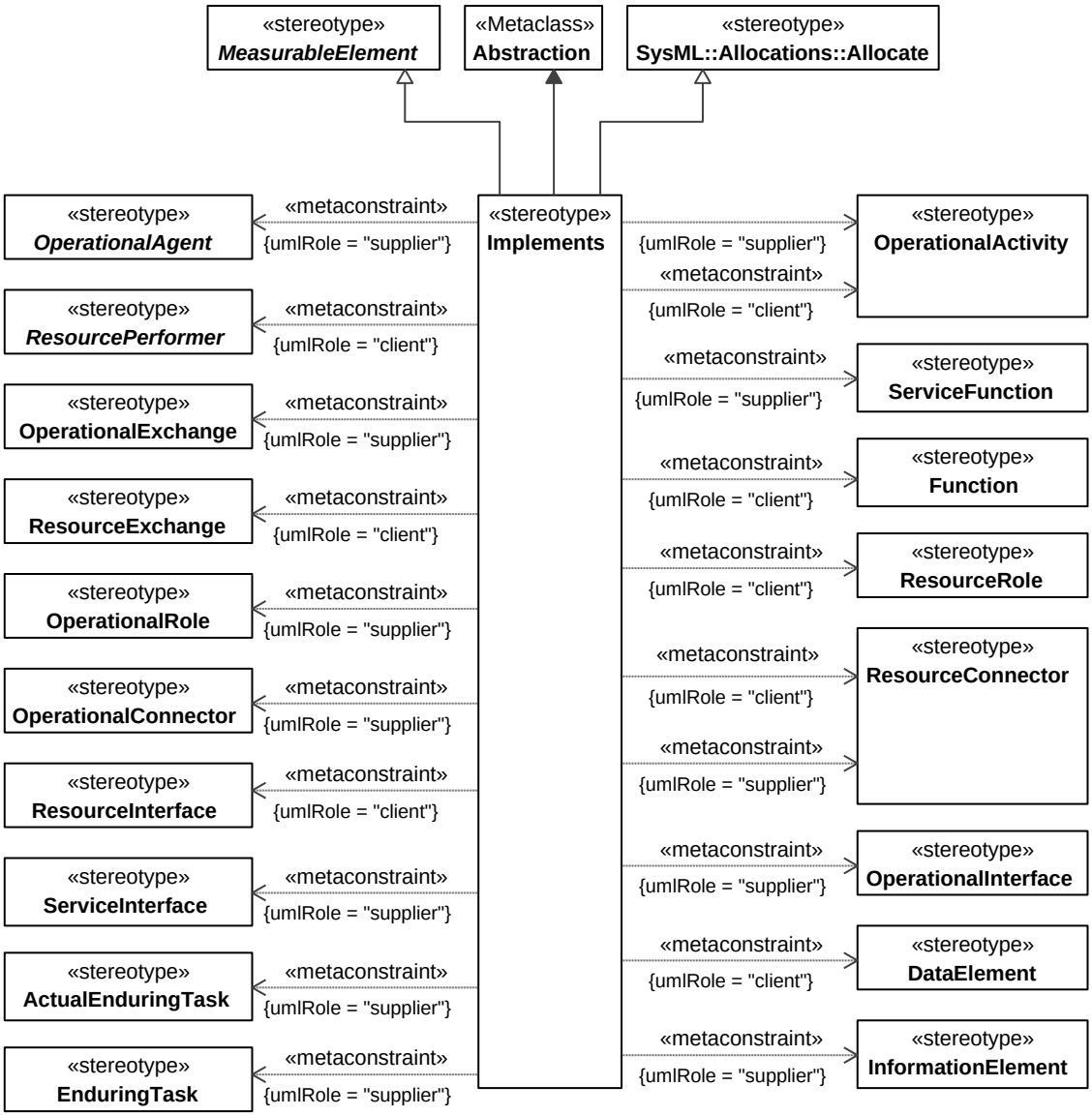

Figure 38 – Implements

Constraints

[1] Implements.client

In case of value for Implements.supplier is stereotyped:
a. by any of specializations of «OperationalAgent», values for the client metaproperty must be stereotyped by any of specializations of «ResourcePerformer»,
b. «OperationalActivity» or its specializations, values for the client metaproperty must be stereotyped «Function» or its specializations,
c. «ServiceFunction» or its specializations, values for the client metaproperty must be stereotyped «Function» or its specializations,
d. «ServiceInterface» or its specializations, values for the client metaproperty must be stereotyped «ResourceInterface» or its specializations,
e. «OperationalInterface» or its specializations, values for the client metaproperty must be stereotyped «ResourceInterface» or its specializations,
f. «OperationalConnector» or its specializations, values for the client metaproperty must be stereotyped «ResourceConnector» or its specializations,
g. «OperationalExchange» or its specializations, values for the client metaproperty must be stereotyped «ResourceExchange» or its specializations,
g. «OperationalRole» or its specializations, values for the client metaproperty must be

	stereotyped «ResourceRole» or its specializations, h. «ResourceConnector» or its specializations, values for the client metaproperty must be stereotyped «ResourceConnector» or its specializations, i. «ActualEnduringTask» or its specializations, values for the client metaproperty must be stereotyped «OperationalActivity» or its specializations, j. «InformationElement» or its specializations, values for the client metaproperty must be stereotyped «DataElement» or its specializations.
[2] Implements.supplier	In case of value for Implements.client is stereotyped: a. by any of specializations of «ResourcePerformer», values for the supplier metaproperty must be stereotyped by any of specializations of «OperationalAgent», b. «Function» or its specializations, values for the supplier metaproperty must be stereotyped «OperationalActivity», «ServiceFunction» or their specializations, c. «ResourceInterface» or its specializations, values for the supplier metaproperty must be stereotyped «ServiceInterface» or its specializations, d. «ResourceInterface» or its specializations, values for the supplier metaproperty must be stereotyped «OperationalInterface» or its specializations, e. «ResourceConnector» or its specializations, values for the supplier metaproperty must be stereotyped «OperationalConnector», «ResourceConnector» or their specializations, f. «ResourceExchange» or its specializations, values for the supplier metaproperty must be stereotyped «OperationalExchange» or its specializations, g. «ResourceRole» or its specializations, values for the supplier metaproperty must be stereotyped «OperationalRole» or its specializations, h. «OperationalActivity» or its specializations, values for the supplier metaproperty must be stereotyped «ActualEnduringTask» or its specializations, i. «DataElement» or its specializations, values for the supplier metaproperty must be stereotyped «InformationElement» or its specializations.

7.1.4 UAF::Strategic

Stakeholders: Capability Portfolio Managers.
Concerns: capability management process.
Definition: describe capability taxonomy, composition, dependencies and evolution.

7.1.4.1 UAF::Strategic::Taxonomy

Contains the elements that contribute to the Strategic Taxonomy Viewpoint.

ActualEnterprisePhase

Package: Taxonomy

isAbstract: No

Generalization: ActualPropertySet, CapableElement, Achiever

Extension: InstanceSpecification
Description

An ActualState that describes the phase of an Enterprise endeavour.

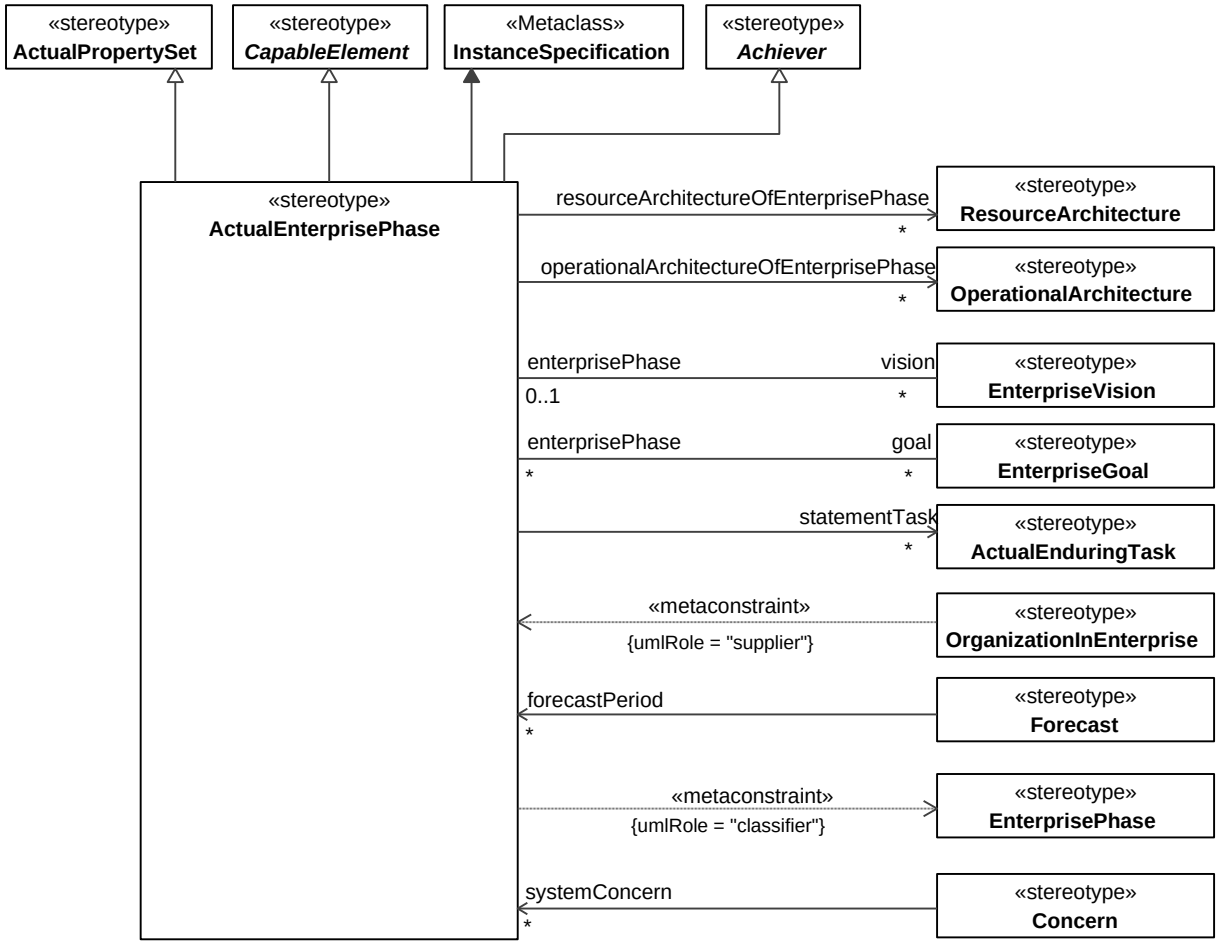

Figure 39 – ActualEnterprisePhase

Associations

goal : EnterpriseGoal[*]	The Goal towards which this Phase is directed and is in support of.
operationalArchitectureOfEnterprisePhase : OperationalArchitecture[*]	Relates an ActualEnterprisePhase to its relevant OperationalArchitecture.
resourceArchitectureOfEnterprisePhase : ResourceArchitecture[*]	Relates an ActualEnterprisePhase to its relevant ResourceArchitecture.
statementTask : ActualEnduringTask[*]	Relates the ActualEnterprisePhase to the ActualEnduringTasks that are intended to be implemented during that phase.
vision : EnterpriseVision[*]	The Vision towards which this Phase is directed and is in support of.

Constraints

[1] ActualEnterprisePhase.classifier	Value for the classifier metaproperty must be stereotyped by «EnterprisePhase» or its specializations.
[2] ActualEnterprisePhase.start/endDate	Must fall within the start and end dates of the enclosing ActualEnterprisePhase having this ActualEnterprisePhase set as a value for a slot.

Capability

Package: Taxonomy

isAbstract: No

Generalization: PropertySet, Desirer, Block

Extension: Class
Description

A high level specification of the enterprise's ability to execute a specified course of action.

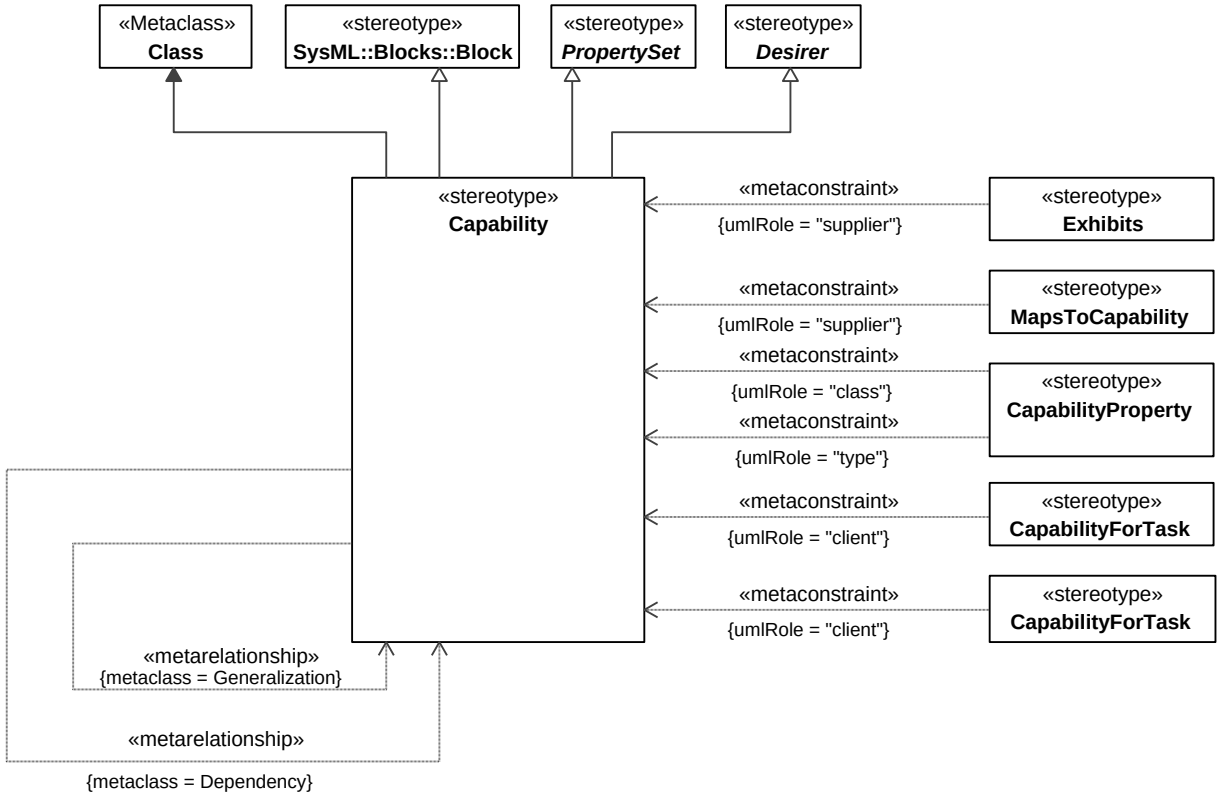

Figure 40 – Capability

EnterpriseGoal

Package: Taxonomy

isAbstract: No

Generalization: PropertySet, Requirement

Extension: Class
Description

A statement about a state or condition of the enterprise to be brought about or sustained through appropriate Means. An EnterpriseGoal amplifies an EnterpriseVision that is, it indicates what must be satisfied on a continuing basis to effectively attain the EnterpriseVision. http://www.omg.org/spec/BMM/1.3/

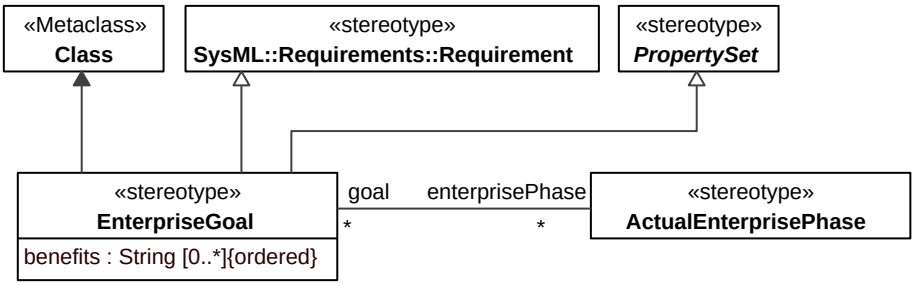

Figure 41 – EnterpriseGoal

Attributes

benefits : String[0..*] A description of the usefulness of the Goal in terms of why the state or condition of the Enterprise is worth attaining.

Associations

enterprisePhase : ActualEnterprisePhase[*] Relates the EnterpriseGoal to the ActualEnterprisePhase in which the EnterpriseGoal is attained.

EnterprisePhase

Package: Taxonomy

isAbstract: No

Generalization: PropertySet, Block, CapableElement

Extension: Class

Description

A current or future state of the wholeLifeEnterprise or another EnterprisePhase.

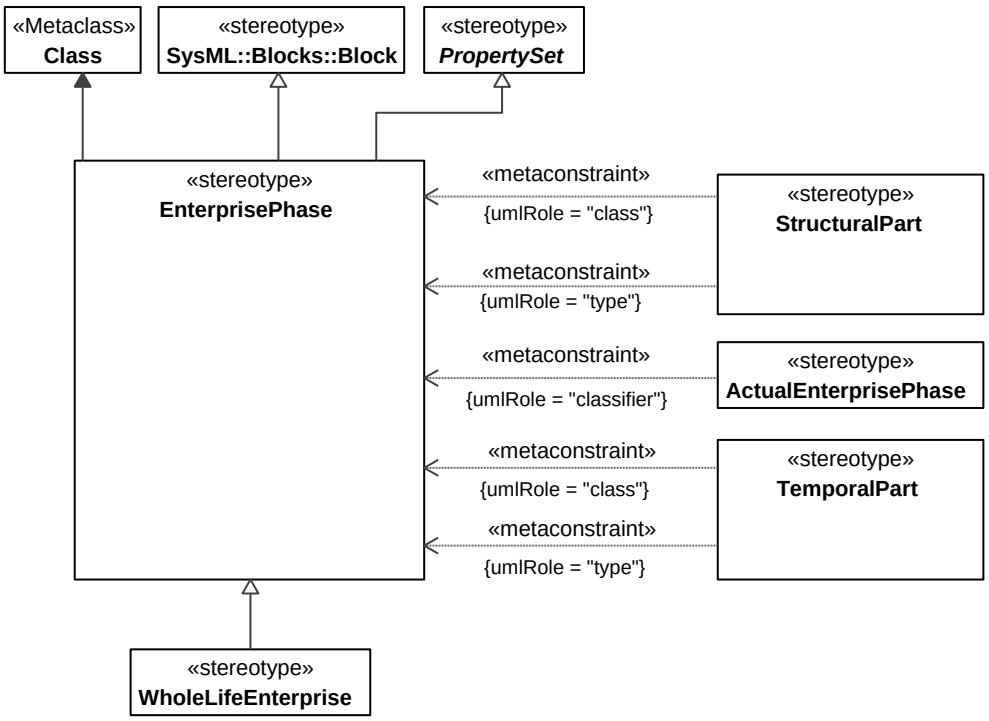

Figure 42 – EnterprisePhase

EnterpriseVision

Package: Taxonomy

isAbstract: No

Generalization: PropertySet, Block

Extension: Class
Description

A Vision describes the future state of the enterprise, without regard to how it is to be achieved.
http://www.omg.org/spec/BMM/1.3/

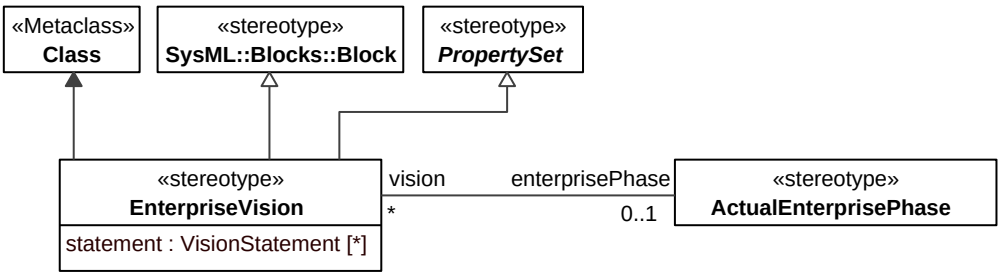

Figure 43 – EnterpriseVision

Attributes

statement : VisionStatement[*] A description of the Vision.

Associations

enterprisePhase : ActualEnterprisePhase[0..1] Relates the EnterpriseVision to the ActualEnterprisePhase in which the EnterpriseVision is expected to be realized.

VisionStatement

Package: Taxonomy

isAbstract: No

Generalization: MeasurableElement

Extension: Comment
Description

A type of comment that describes the future state of the enterprise, without regard to how it is to be achieved.
http://www.omg.org/spec/BMM/1.3/

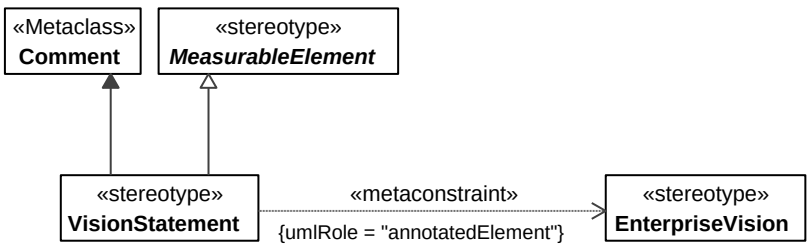

Figure 44 – VisionStatement

Constraints

[1] VisionStatement.annotatedElement Values for annotatedElement metaproperty must be stereotyped «EnterpriseVision» or its specializations.

WholeLifeEnterprise

Package: Taxonomy

isAbstract: No

Generalization: EnterprisePhase

Extension: Class
Description

A WholeLifeEnterprise is a purposeful endeavor of any size involving people, organizations and supporting systems. It is made up of TemporalParts and StructuralParts.

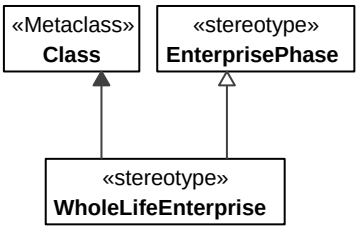

Figure 45 – WholeLifeEnterprise

7.1.4.2 UAF::Strategic::Structure

Contains the elements that contribute to the Strategic Structure Viewpoint.

CapabilityProperty

Package: Structure

isAbstract: No

Generalization: MeasurableElement

Extension: Property
Description

Property of a Capability typed by another Capability, enabling whole-part relationships and structures.

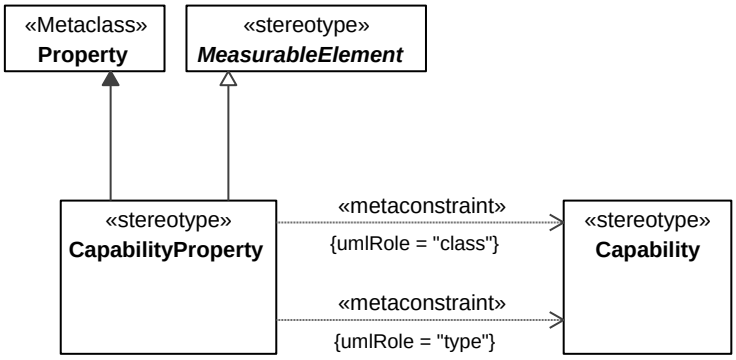

Figure 46 – CapabilityProperty

Constraints

[1] CapabilityProperty.class	Value for class metaproperty must be stereotyped «Capability» or its specializations.
[2] CapabilityProperty.type	Value for type metaproperty must be stereotyped «Capability» or its specializations.

StructuralPart

Package: Structure

isAbstract: No

Generalization: MeasurableElement

Extension: Property
Description

Usage of an EnterprisePhase in the context of another EnterprisePhase. It asserts that one EnterprisePhase is a spatial part of another. Creates a whole-part relationship that represents the structure of the EnterprisePhase.

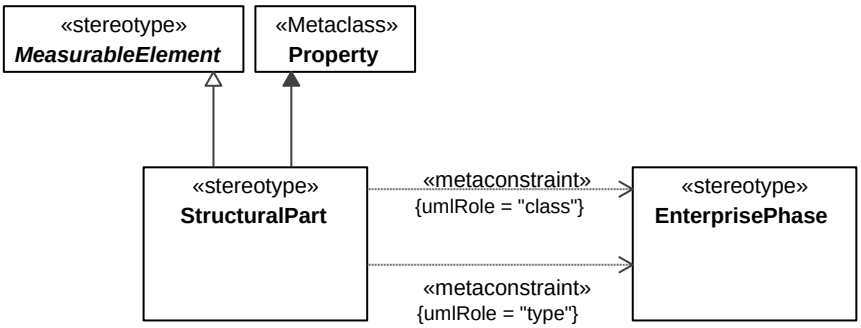

Figure 47 – StructuralPart

Constraints

[1] StructuralPart.class	Value for class metaproperty must be stereotyped «EnterprisePhase» or its specializations.
[2] StructuralPart.type	Value for type metaproperty must be stereotyped «EnterprisePhase» or its specializations.

TemporalPart

Package: Structure

isAbstract: No

Generalization: MeasurableElement

Extension: Property
Description

Usage of an EnterprisePhase in the context of another EnterprisePhase. It asserts that one EnterprisePhase is a spatial part of another. Creates a whole-part relationship that represents the temporal structure of the EnterprisePhase.

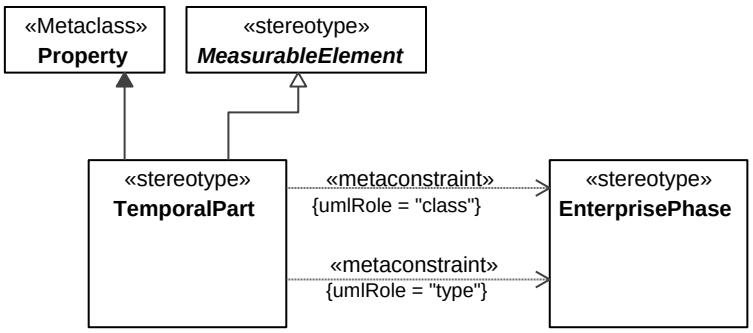

Figure 48 – TemporalPart

Constraints

[1] TemporalPart.class	Value for class metaproperty must be stereotyped «EnterprisePhase» or its specializations.
[2] TemporalPart.type	Value for type metaproperty must be stereotyped «EnterprisePhase» or its specializations.

7.1.4.3 UAF::Strategic::Processes

Contains the elements that contribute to the Strategic Proceses Viewpoint.

ActualEnduringTask

Package: Processes

isAbstract: No

Generalization: CapableElement, ActualPropertySet

Extension: InstanceSpecification
Description

An actual undertaking recognized by an enterprise as being essential to achieving its goals - i.e. a strategic specification of what the enterprise does.

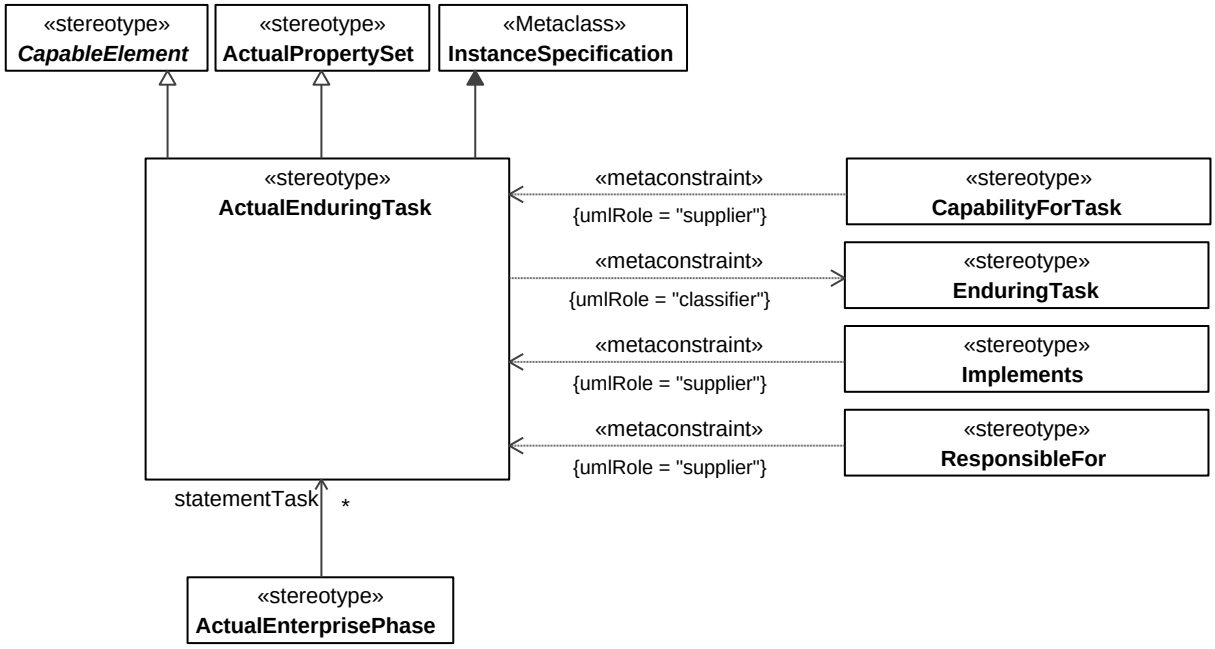

Figure 49 – ActualEnduringTask

Constraints

[1] ActualEnduringTask.classifier	Value for the classifier metaproperty must be stereotyped by «EnduringTask» or its specializations.

CapabilityForTask

Package: Processes

isAbstract: No

Generalization: MeasurableElement, Allocate

Extension: Abstraction
Description

An abstraction relationship that asserts that a Capability is required in order for an Enterprise to conduct a phase of an EnduringTask.

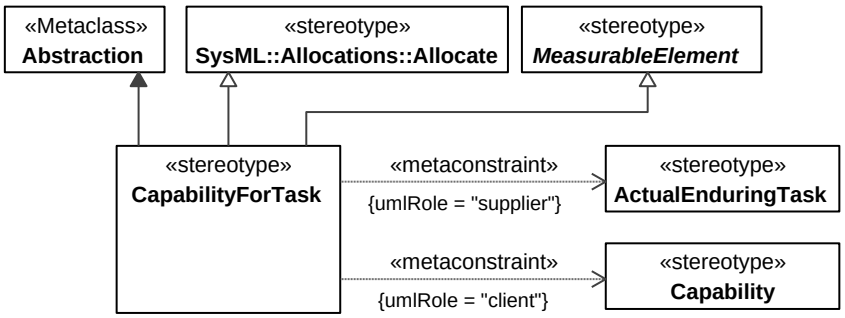

Figure 50 – CapabilityForTask

Constraints

[1] CapabilityForTask.client	Value for the client metaproperty must be stereotyped «Capability» or its specializations.
[2] CapabilityForTask.supplier	Value for the supplier metaproperty must be stereotyped «ActualEnduringTask» or its specializations.

EnduringTask

Package: Processes

isAbstract: No

Generalization: PropertySet, Block

Extension: Class
Description

A type of template behavior recognized by an enterprise as being essential to achieving its goals - i.e. a template for a strategic specification of what the enterprise does.

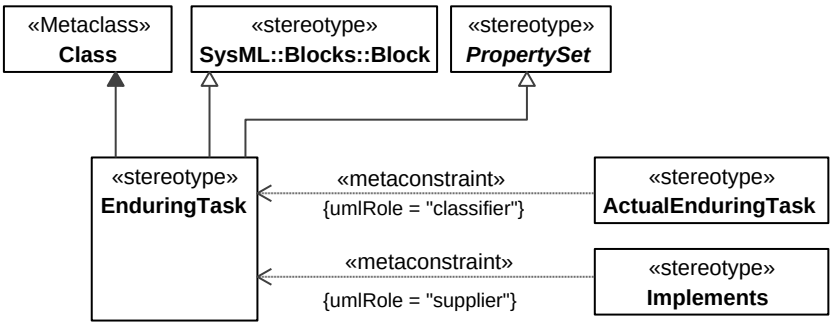

Figure 51 – EnduringTask

7.1.4.4 UAF::Strategic::States

Contains the elements that contribute to the Strategic States Viewpoint.

AchievedEffect

Package: States

isAbstract: No

Generalization: MeasurableElement

Extension: Dependency
Description

A dependency relationship that exists between an ActualState (e.g., observed/measured during testing) of an element that attempts to achieve a DesiredEffect and an Achiever.

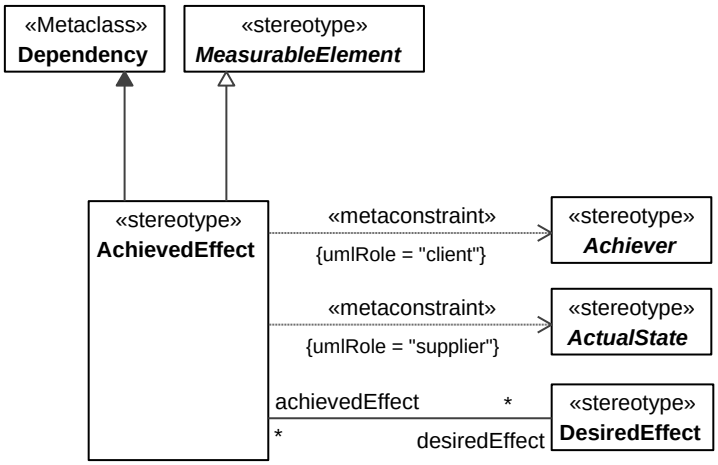

Figure 52 – AchievedEffect

Associations

desiredEffect : DesiredEffect[*]	Relates the effect that is achieved with the originally expected DesirectEffect. Providing a means of comparison, between the expectation of the desirer and the actual result.

Constraints

[1] AchievedEffect.client	Value for the client metaproperty must be stereotyped by the specialization of «Achiever».
[2] AchievedEffect.supplier	Value for the supplier metaproperty must be stereotyped by the specialization of «ActualState».

Achiever

Package: States

isAbstract: Yes

Generalization: UAFElement

Extension: InstanceSpecification

Description

An ActualResource, ActualProject or ActualEnterprisePhase that can deliver a DesiredEffect.

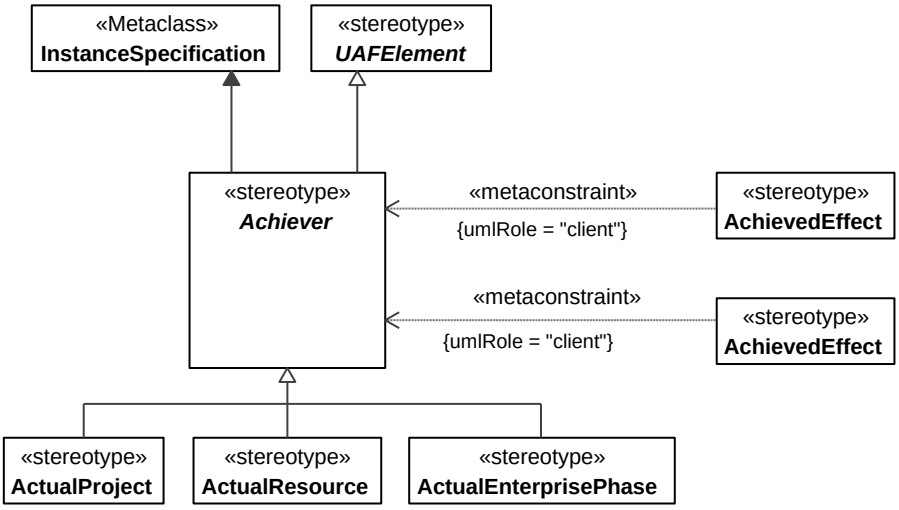

Figure 53 – Achiever

DesiredEffect

Package: States

isAbstract: No

Generalization: MeasurableElement

Extension: Dependency

Description

A dependency relationship relating the Desirer (a Capability or OrganizationalResource) to an ActualState.

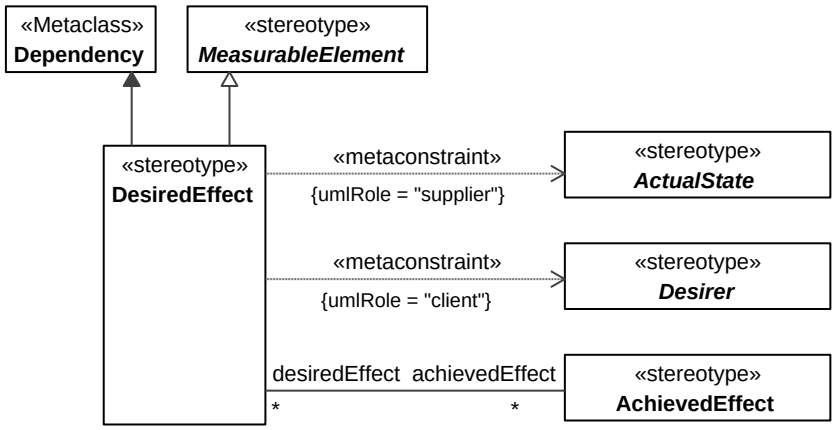

Figure 54 – DesiredEffect

Associations

achievedEffect : AchievedEffect[*]

Constraints

[1] DesiredEffect.client Value for the client metaproperty must be stereotyped a specialization of «Desirer».

[2] DesiredEffect.supplier Value for the supplier metaproperty must be stereotyped a specialization of «ActualState».

Desirer

Package: States

isAbstract: Yes

Generalization: UAFElement

Extension: Class

Description

Abstract element used to group architecture elements that might desire a particular effect.

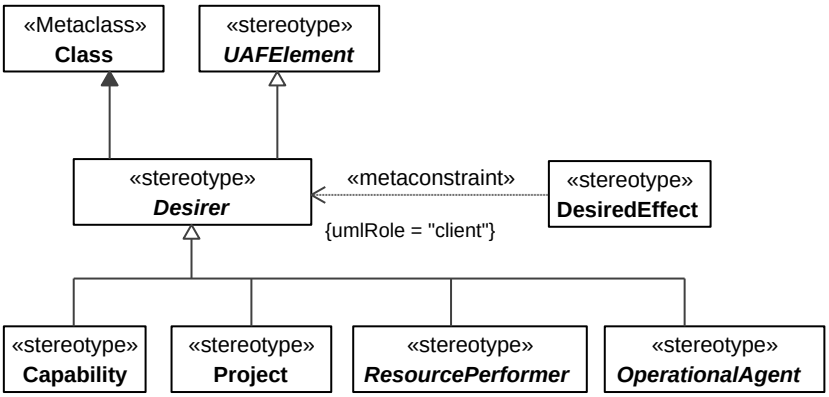

Figure 55 – Desirer

7.1.4.5 UAF::Strategic::Traceability

Contains the elements that contribute to the Strategic Traceability Viewpoint.

Exhibits

Package: Traceability

isAbstract: No

Generalization: MeasurableElement, Allocate

Extension: Abstraction

Description

An abstraction relationship that exists between a CapableElement and a Capability that it meets under specific environmental conditions.

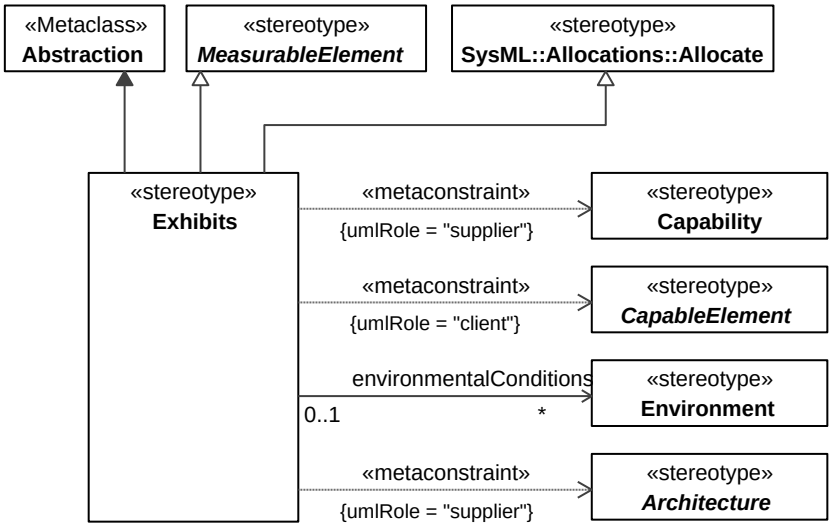

Figure 56 – Exhibits

Associations

environmentalConditions : Environment[*] Defines the environmental conditions constraining the way that a Capability is exhibited.

Constraints

[1] Exhibits.client Value for the client metaproperty must be stereotyped a specialization of «CapableElement».

[2] Exhibits.supplier Value for the supplier metaproperty must be stereotyped «Capability».

MapsToCapability

Package: Traceability

isAbstract: No

Generalization: MeasurableElement, Allocate

Extension: Abstraction

Description

An Abstraction relationship denoting that an Activity contributes to providing a Capability.

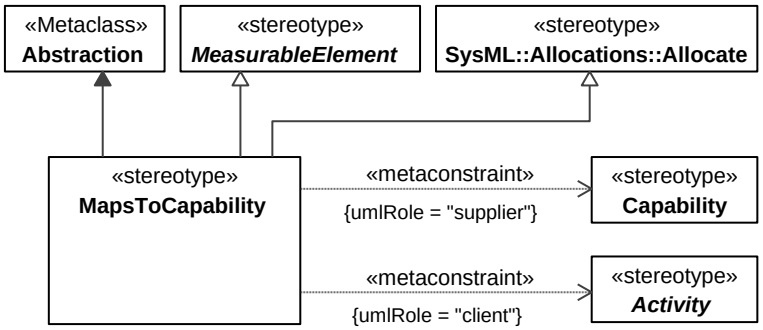

Figure 57 – MapsToCapability

Constraints

[1] MapsToCapability.client Value for the client metaproperty must be stereotyped a specialization of «Activity».

[2] MapsToCapability.supplier Value for the supplier metaproperty must be stereotyped «Capability».

OrganizationInEnterprise

Package: Traceability

isAbstract: No

Generalization: MeasurableElement, Allocate

Extension: Abstraction

Description

An abstraction relationship relating an ActualOrganization to an ActualEnterprisePhase to denote that the ActualOrganization plays a role or is a stakeholder in an ActualEnterprisePhase.

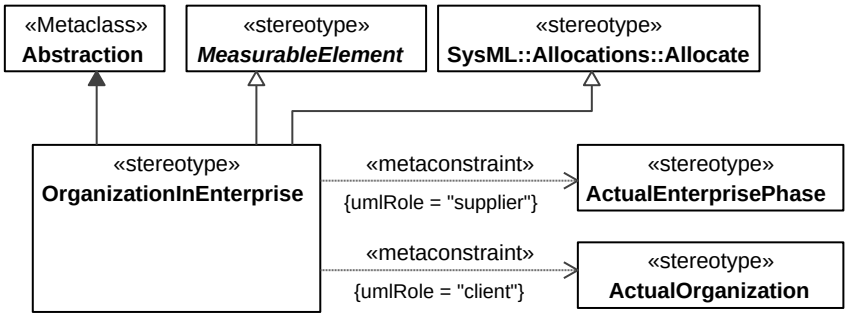

Figure 58 – OrganizationInEnterprise

Constraints

[1] OrganizationInEnterprise.client Value for the client metaproperty must be stereotyped «ActualOrganization» or its specializations.

[2] OrganizationInEnterprise.supplier Value for the supplier metaproperty must be stereotyped «ActualEnterprisePhase» or its specializations.

7.1.5 UAF::Operational

Stakeholders: Business Architects, Executives.
Concerns: illustrate the Logical Architecture of the enterprise.
Definition: describe the requirements, operational behavior, structure, and exchanges required to support (exhibit) capabilities. Defines all operational elements in an implementation/solution independent manner.

7.1.5.1 UAF::Operational::Taxonomy

Contains the elements that contribute to the Operational Taxonomy Viewpoint.

ArbitraryConnector

Package: Taxonomy

isAbstract: No

Generalization: MeasurableElement

Extension: Dependency

Description

Represents a visual indication of a connection used in high level operational concept diagrams.

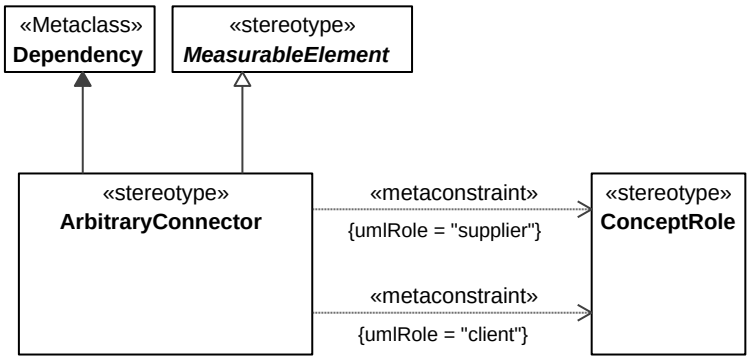

Figure 59 – ArbitraryConnector

Constraints

[1] ArbitraryConnector.client	The value for client metaproperty has to be stereotyped «ConceptRole» or its specializations.
[2] ArbitraryConnector.supplier	The value for supplier metaproperty has to be stereotyped «ConceptRole» or its specializations.

ConceptItem

Package: Taxonomy

isAbstract: Yes

Generalization: UAFElement

Extension: Element

Description

Abstract, an item which may feature in a HighLevelOperationalConcept.

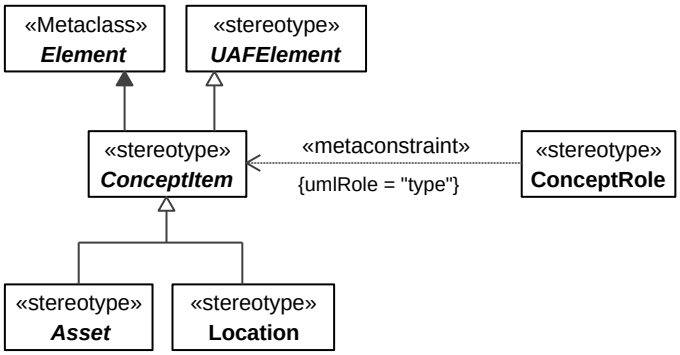

Figure 60 – ConceptItem

ConceptRole

Package: Taxonomy

isAbstract: No

Generalization: MeasurableElement

Extension: Property
Description

Usage of a ConceptItem in the context of a HighLevelOperationalConcept.

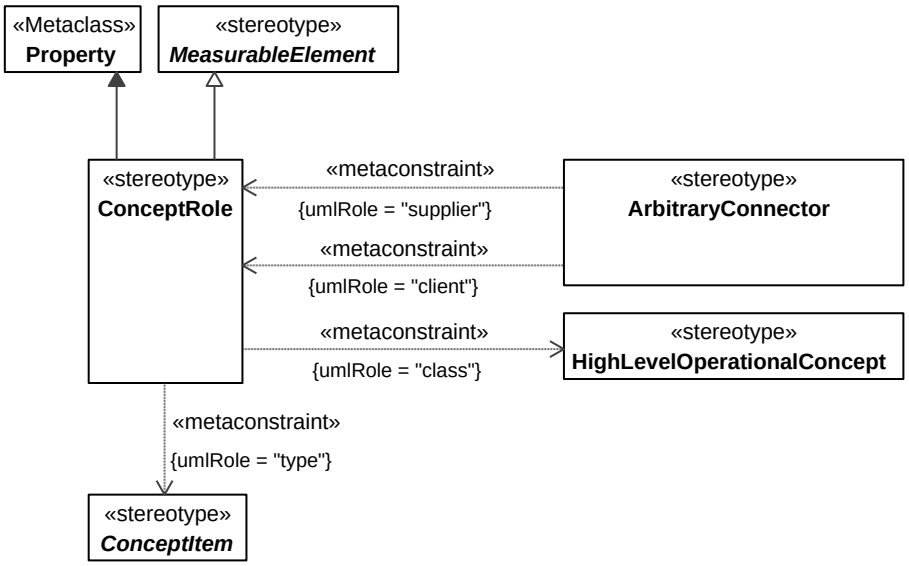

Figure 61 – ConceptRole

Constraints

[1] ConceptRole.class	Value for the class metaproperty must be stereotyped «HighLevelOperationalConcept» or its specializations.
[2] ConceptRole.type	Value for the type metaproperty must be stereotyped by a specialization of «ConceptItem».

HighLevelOperationalConcept

Package: Taxonomy

isAbstract: No

Generalization: PropertySet, Block

Extension: Class
Description

Describes the Resources and Locations required to meet an operational scenario from an integrated systems point of view. It is used to communicate overall quantitative and qualitative system characteristics to stakeholders.

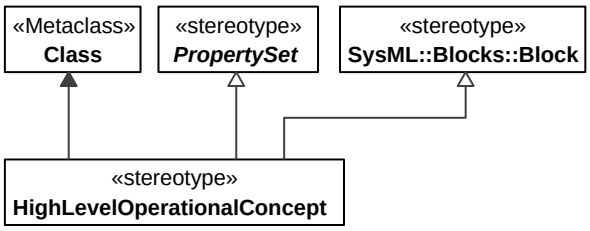

Figure 62 – HighLevelOperationalConcept

7.1.5.2 UAF::Operational::Structure

Contains the elements that contribute to the Operational Structure Viewpoint.

KnownResource

Package: Structure

isAbstract: No

Generalization: OperationalPerformer, ResourcePerformer

Extension: Class
Description

Asserts that a known ResourcePerformer plays a part in the LogicalArchitecture.

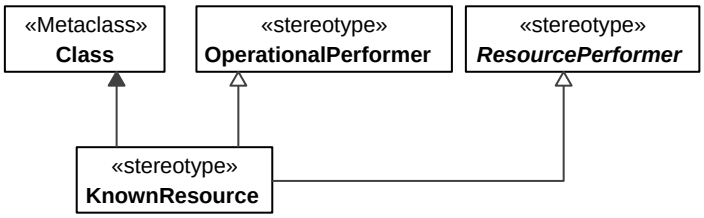

Figure 63 – KnownResource

OperationalAgent

Package: Structure

isAbstract: Yes

Generalization: Asset, SubjectOfOperationalConstraint, CapableElement, Desirer

Extension: Class
Description

An abstract element grouping LogicalArchitecture and OperationalPerformer.

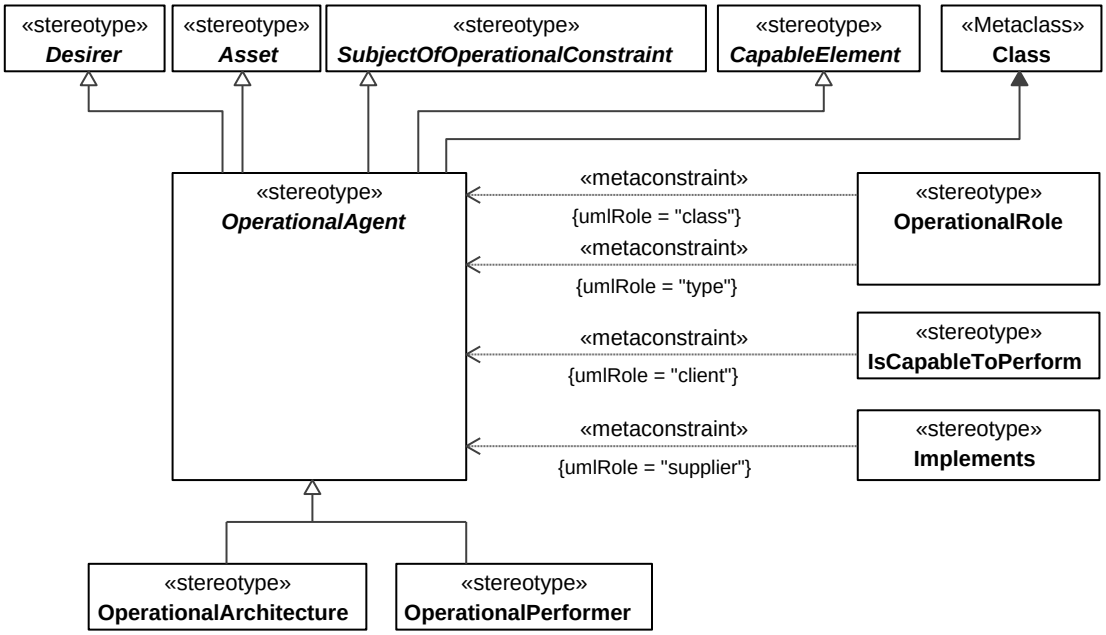

Figure 64 – OperationalAgent

OperationalArchitecture

Package: Structure

isAbstract: No

Generalization: OperationalAgent, Architecture

Extension: Class
Description

An element used to denote a model of the Architecture, described from the Operational perspective.

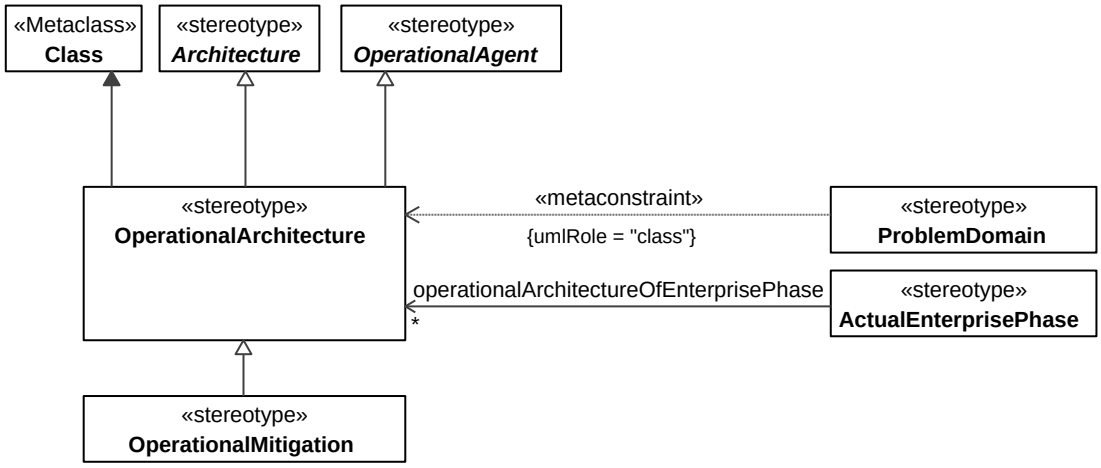

Figure 65 – OperationalArchitecture

OperationalMethod

Package: Structure

isAbstract: No

Generalization: MeasurableElement

Extension: Operation
Description

A behavioral feature of a OperationalPerformer whose behavior is specified in an OperationalActivity.

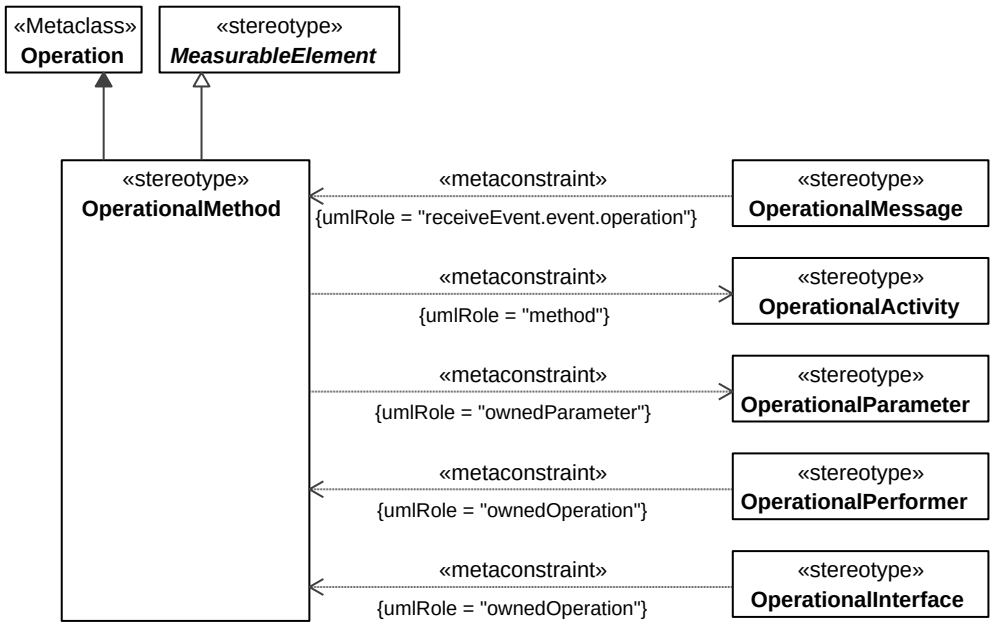

Figure 66 – OperationalMethod

Constraints

[1] OperationalMethod.method Value for the method metaproperty must be stereotyped

«OperationalActivity» or its specializations.

[2] OperationalMethod.ownedParameter	The values for the ownedParameter metaproperty must be stereotyped «OperationalParameter» or its specializations.

OperationalParameter

Package: Structure

isAbstract: No

Generalization: MeasurableElement

Extension: Parameter

Description

An element that represents inputs and outputs of an OperationalActivity. It is typed by an OperationalExchangeItem.

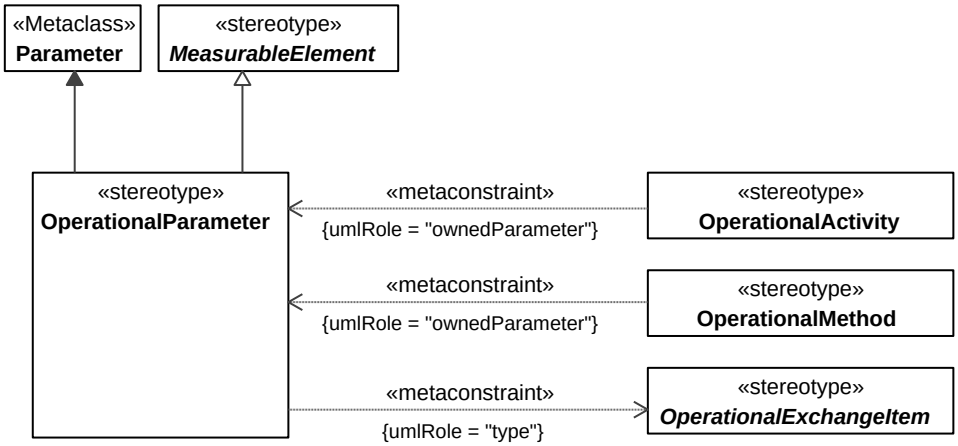

Figure 67 – OperationalParameter

Constraints

[1] OperationalParameter.type	Value for the type metaproperty must be stereotyped by specialization of «OperationalExchangeItem».

OperationalPerformer

Package: Structure

isAbstract: No

Generalization: OperationalAgent

Extension: Class

Description

A logical agent that IsCapableToPerform OperationalActivities which produce, consume and process Resources.

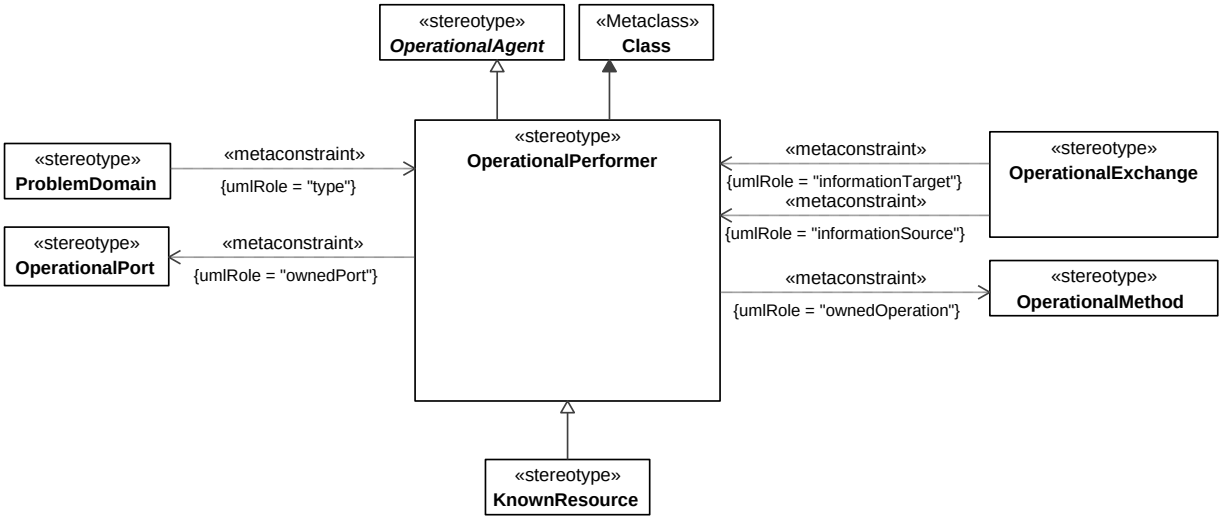

Figure 68 – OperationalPerformer

Constraints

[1] OperationalPerformer.isCapableToPerform	Is capable to perform only «OperationalActivity» elements or its specializations.
[2] OperationalPerformer.ownedOperation	Values for the ownedOperation metaproperty must be stereotyped «OperationalMethod» or its specializations.
[3] OperationalPerformer.ownedPort	Values for the ownedPort metaproperty must be stereotyped «OperationalPort» or its specializations.

OperationalPort

Package: Structure

isAbstract: No

Generalization: MeasurableElement, ProxyPort

Extension: Port

Description

Usage of a OperationalPerformer or LogicalArchitecture in the context of another OperationalPerformer or LogicalArchitecture. Creates a whole-part relationship.

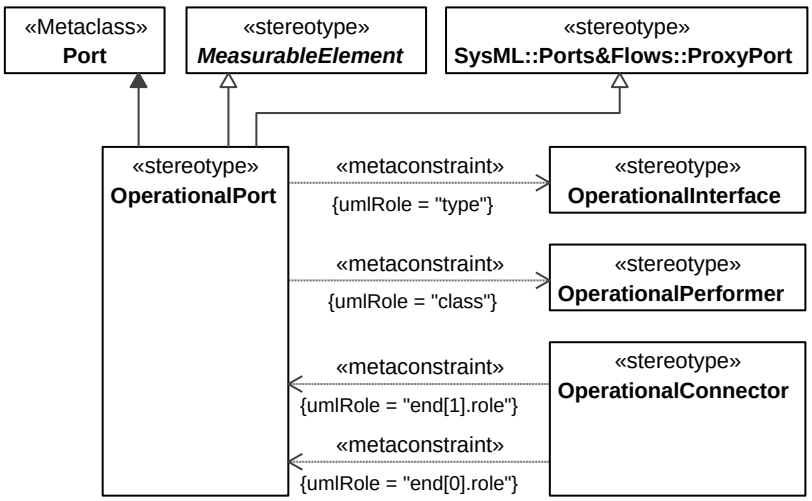

Figure 69 – OperationalPort

Constraints

[1] OperationalPort.class	Value for class metaproperty must be stereotyped «OperationalPerformer» or its specializations.
[2] OperationalPort.type	Value for type metaproperty must be stereotyped «OperationalInterface» or its specializations.

OperationalRole

Package: Structure

isAbstract: No

Generalization: MeasurableElement, LocationHolder, SubjectOfSecurityConstraint, AssetRole

Extension: Property

Description

Usage of a OperationalPerformer or OperationalArchitecture in the context of another OperationalPerformer or OperationalArchitecture. Creates a whole-part relationship.

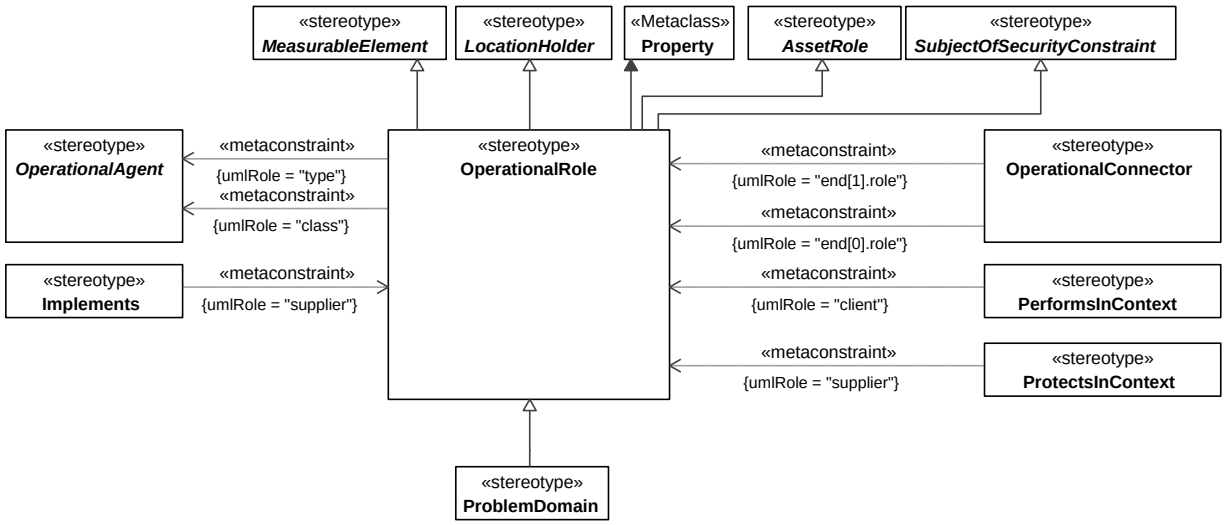

Figure 70 – OperationalRole

Constraints

[1] OperationalRole.class	Value for class metaproperty must be stereotyped by a specialization of «OperationalAgent».
[2] OperationalRole.type	Value for type metaproperty must be stereotyped by a specialization of «OperationalAgent».

ProblemDomain

Package: Structure

isAbstract: No

Generalization: OperationalRole

Extension: Property

Description

A property associated with a logical architecture, used to specify the scope of the problem.

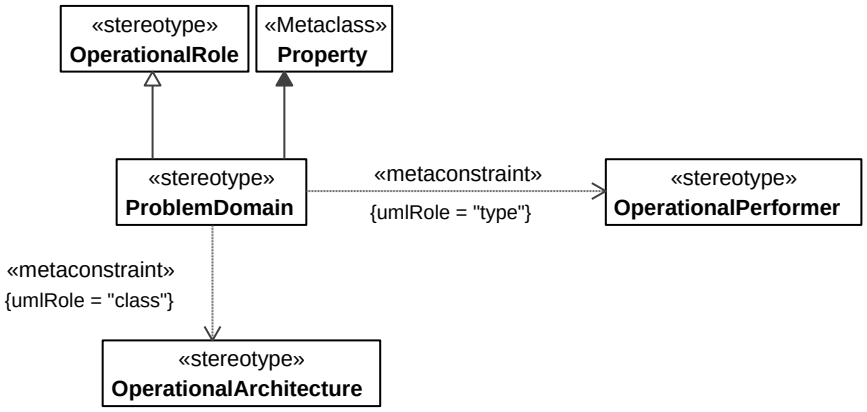

Figure 71 – ProblemDomain

Constraints

[1] ProblemDomain.class	Value for the class metaproperty must be stereotyped «OperationalArchitecture» or its specializations.
[2] ProblemDomain.type	Value for the type metaproperty must be stereotyped «OperationalPerformer» or its specializations.

7.1.5.3 UAF::Operational::Connectivity

Contains the elements that contribute to the Operational Connectivity Viewpoint.

OperationalConnector

Package: Connectivity

isAbstract: No

Generalization: MeasurableElement

Extension: Connector

Description

A Connector that goes between OperationalRoles representing a need to exchange Resources. It can carry a number of OperationalExchanges.

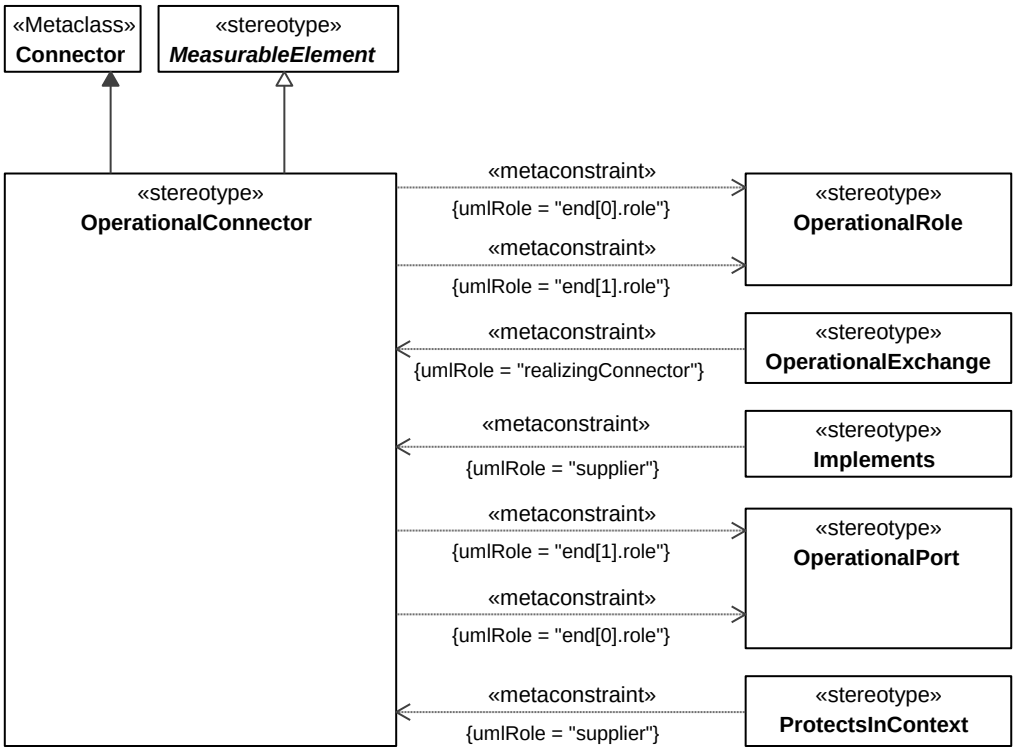

Figure 72 – OperationalConnector

Constraints

[1] OperationalConnector.end	The value for the role metaproperty for the owned ConnectorEnd must be stereotype «OperationalRole»/«OperationalPort» or its specializations.

OperationalExchange

Package: Connectivity

isAbstract: No

Generalization: Exchange, SubjectOfOperationalConstraint

Extension: InformationFlow

Description

Asserts that a flow can exist between OperationalPerformers (i.e. flows of information, people, materiel, or energy).

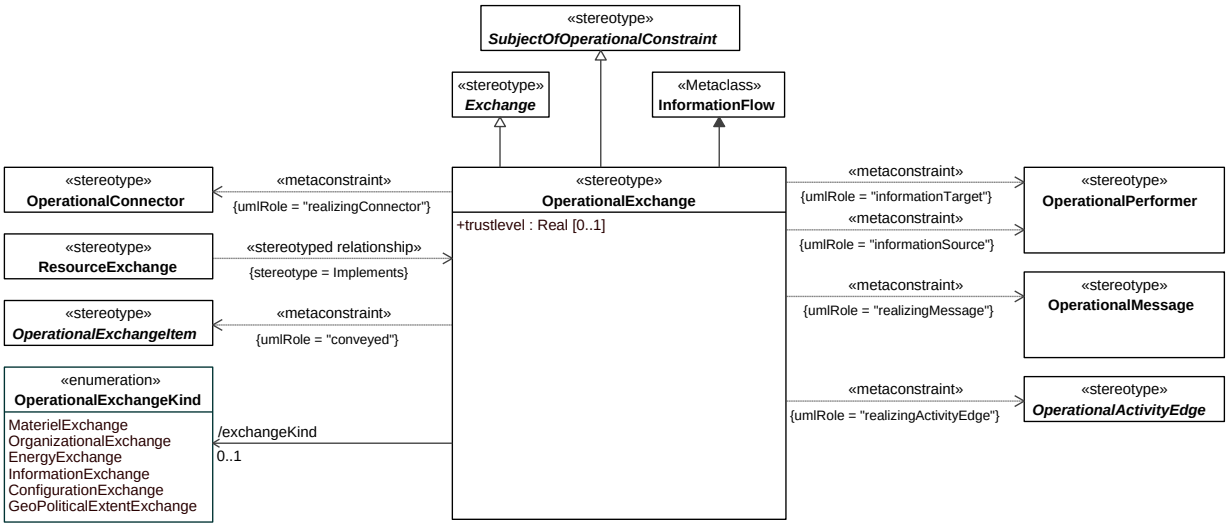

Figure 73 – OperationalExchange

Attributes

trustlevel : Real[0..1] Captures the directional arbitrary level of trust related to an OperationalExchange between two OperationalPerformers.

Associations

exchangeKind : OperationalExchangeKind[0..1] Captures the kind of Resource being exchanged.

Constraints

[1] OperationalExchange.conveyed	In case of OperationalExchange.operationalExchangeKind: = InformationExchange, the conveyed element must be stereotyped «InformationElement» or its specializations, = MaterielExchange, the conveyed element must be stereotyped «ResourceArtifact» or its specializations, = EnergyExchange, the conveyed element must be stereotyped «NaturalResource» or its specializations, = OrganizationalExchange, the conveyed element must be stereotyped «OrganizationalResource» or its specializations, = ConfigurationExchange, the conveyed element must be stereotyped «CapabilityConfiguration» or its specializations, or = GeoPoliticalExtentExchange, the conveyed element must be stereotyped «GeoPoliticalExtentType» or its specializations.
[2] OperationalExchange.informationSource	Value for informationSource metaproperty has to be stereotyped «OperationalPerformer» or its specializations.
[3] OperationalExchange.informationTarget	Value for informationTarget metaproperty has to be stereotyped «OperationalPerformer» or its specializations.
[4] OperationalExchange.realizingActivityEdge	Value for realizingActivityEdge metaproperty has to be stereotyped by any specialization of «OperationalActivityEdge».
[5] OperationalExchange.realizingConnector	Value for realizingConnector metaproperty has to be stereotyped «OperationalConnector» or its specializations.
[6] OperationalExchange.realizingMessage	Value for realizingMessage metaproperty has to be stereotyped «OperationalMessage» or its specializations.

OperationalExchangeItem

Package: Connectivity

isAbstract: Yes

Generalization: Resource

Description

An abstract grouping for elements that defines the types of elements that can be exchanged between OperationalPerformers and conveyed by an OperationalExchange.

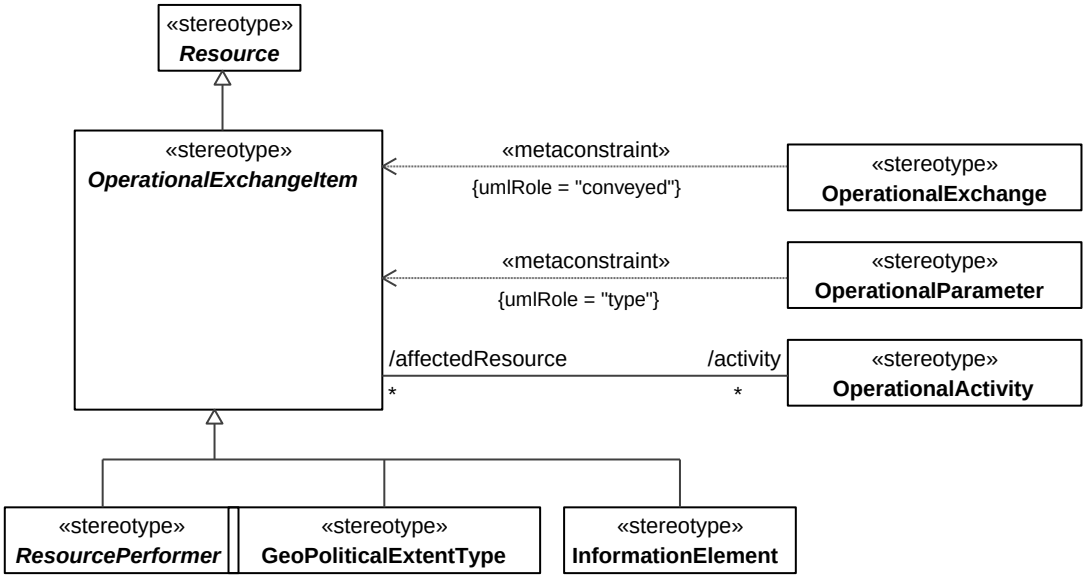

Figure 74 – OperationalExchangeItem

Associations

activity : OperationalActivity[*] A collection of OperationalActivities that consume and/or produce the OperationalExchangeItem internally.

OperationalExchangeKind

Package: Connectivity

isAbstract: No

Description

Enumeration of the possible kinds of operational exchange applicable to an OperationalExchange. Its enumeration literals are:

- MaterielExchange - Indicates that the OperationalExchange associated with the OperationalExchangeKind is a logical flow of materiel (artifacts) between Functions.
- OrganizationalExchange - Indicates that the OperationalExchange associated with the OperationalExchangeKind is a logical flow where human resources (PostTypes, RoleTypes) flow between OperationalPerformers.
- EnergyExchange - Indicates that the OperationalExchange associated with the OperationalExchangeKind is a logical flow where energy is flowed from one OperationalPerformer to another.
- InformationExchange - Indicates that the OperationalExchange associated with the OperationalExchangeKind is a logical flow where information is flowed from one OperationalPerformer to another.
- ConfigurationExchange - Indicates that the OperationalExchange associated with the OperationalExchangeKind is a logical flow where CapabilityConfigurations flow from one OperationalPerformer to another.
- GeoPoliticalExtentExchange - Indicates that the OperationalExchange associated with the OperationalExchangeKind is a logical flow where GeoPoliticalExtentTypes (i.e. Borders) flow from one place to another.

OperationalInterface

Package: Connectivity

isAbstract: No

Generalization: PropertySet, InterfaceBlock

Extension: Class
Description

A declaration that specifies a contract between the OperationalPerformer it is related to, and any other OperationalPerformers it can interact with.

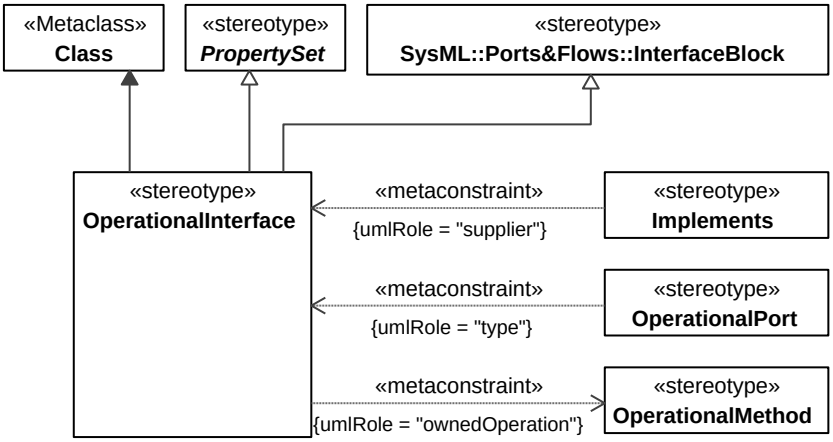

Figure 75 – OperationalInterface

Constraints

[1] OperationalInterface.ownedOperation	Values for the ownedOperation metaproperty must be stereotyped «OperationalMethod» or its specializations.

7.1.5.4 UAF::Operational::Processes

Contains the elements that contribute to the Operational Processes Viewpoint.

OperationalActivity

Package: Processes

isAbstract: No

Generalization: Activity, SubjectOfOperationalConstraint

Extension: Activity
Description

An Activity that captures a logical process, specified independently of how the process is carried out.

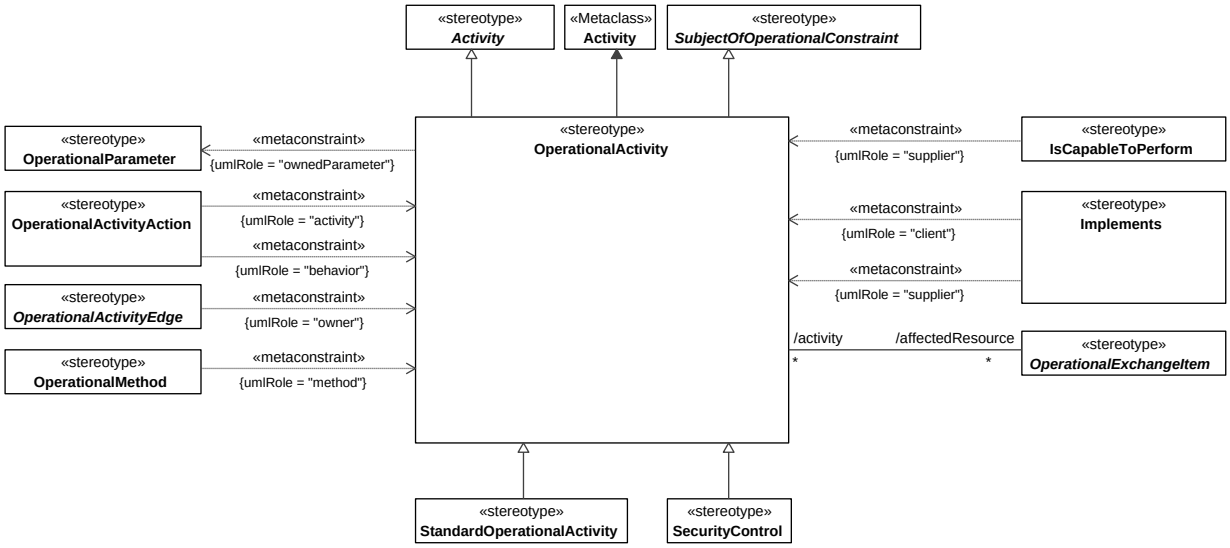

Figure 76 – OperationalActivity

Associations

affectedResource : OperationalExchangeItem[*] A collection of OperationalExchangeItems consumed and produced internally within the OperationalActivity.

Constraints

[1] OperationalActivity.ownedParameter The values for the ownedParameter metaproperty must be stereotyped «OperationalParameter» or its specializations.

OperationalActivityAction

Package: Processes

isAbstract: No

Generalization: MeasurableElement

Extension: CallBehaviorAction

Description

A call of an OperationalActivity in the context of another OperationalActivity.

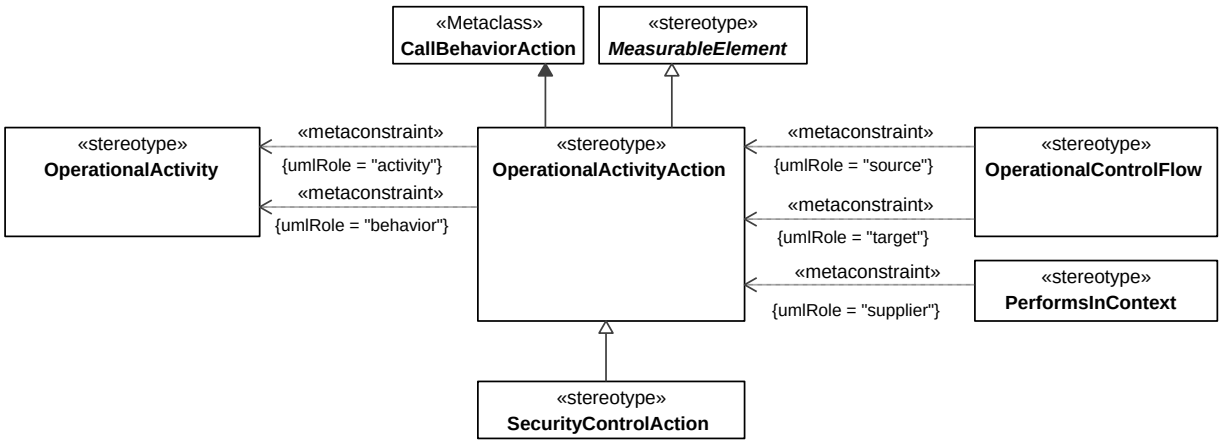

Figure 77 – OperationalActivityAction

Constraints

[1] OperationalActivityAction.activity Value for the activity metaproperty must be stereotyped «OperationalActivity» or its specializations.

[2] OperationalActivityAction.behavior Value for activity metaproperty must be stereotyped «OperationalActivity» or

its specializations.

OperationalActivityEdge

Package: Processes

isAbstract: Yes

Generalization: MeasurableElement

Extension: ActivityEdge

Description

Abstract grouping for OperationalControlFlow and OperationalObjectFlow.

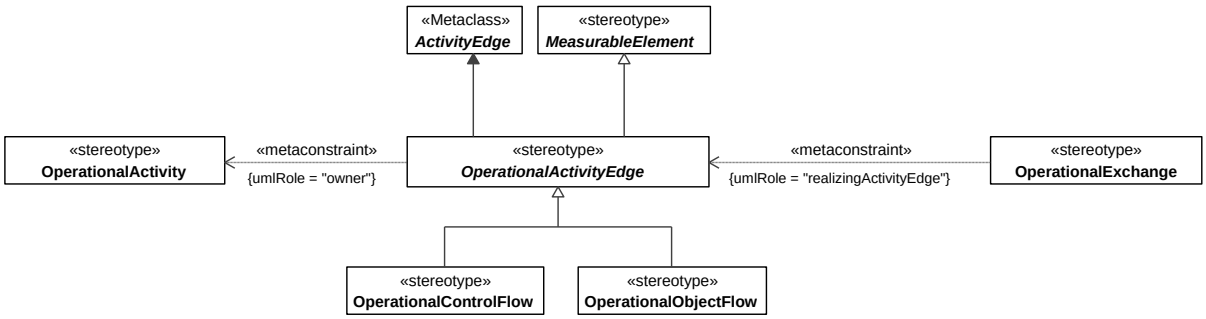

Figure 78 – OperationalActivityEdge

Constraints

[1] OperationalActivityEdge.owner	«OperationalActivityEdge» must be owned directly or indirectly by «OperationalActivity» or its specializations.

OperationalControlFlow

Package: Processes

isAbstract: No

Generalization: OperationalActivityEdge

Extension: ControlFlow

Description

An ActivityEdge that shows the flow of control between OperationalActivityActions.

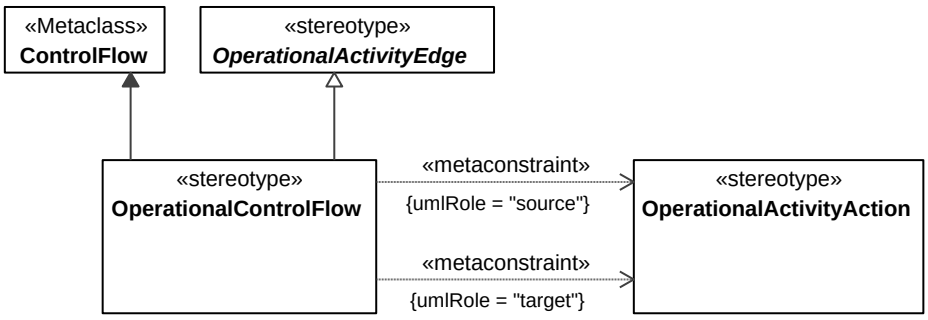

Figure 79 – OperationalControlFlow

Constraints

[1] OperationalControlFlow.source	Value for the source metaproperty must be stereotyped «OperationalActivityAction» or its specializations.
[2] OperationalControlFlow.target	Value for the target metaproperty must be stereotyped «OperationalActivityAction» or its specializations.

OperationalObjectFlow

Package: Processes

isAbstract: No

Generalization: OperationalActivityEdge

Extension: ObjectFlow
Description

An ActivityEdge that shows the flow of Resources (objects/information) between OperationalActivityActions.

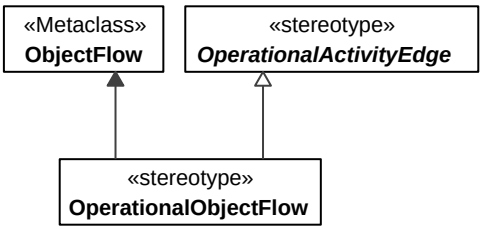

Figure 80 – OperationalObjectFlow

StandardOperationalActivity

Package: Processes

isAbstract: No

Generalization: OperationalActivity

Extension: Activity
Description

A sub-type of OperationalActivity that is a standard operating procedure.

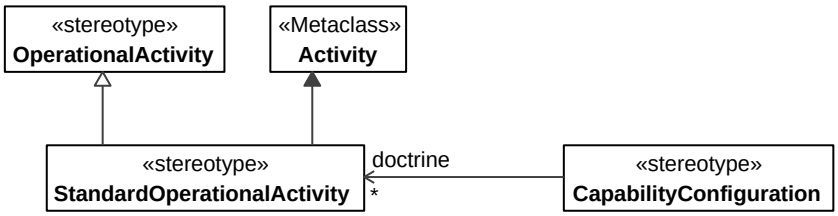

Figure 81 – StandardOperationalActivity

7.1.5.5 UAF::Operational::States

Contains the elements that contribute to the Operational States Viewpoint.

OperationalStateDescription

Package: States

isAbstract: No

Generalization: MeasurableElement

Extension: StateMachine
Description

A state machine describing the behavior of a OperationalPerformer, depicting how the OperationalPerformer responds to various events and the actions.

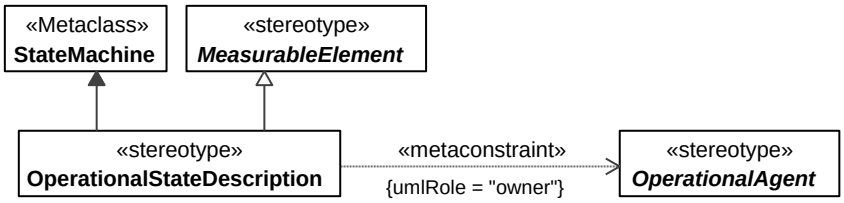

Figure 82 – OperationalStateDescription

Constraints

[1] OperationalStateDescription.owner — Values for the owner metaproperty must be stereotyped with specializations of «OperationalAgent» .

7.1.5.6 UAF::Operational::Interaction Scenarios

Contains the elements that contribute to the Operational Interaction Scenarios Viewpoint.

OperationalMessage

Package: Interaction Scenarios

isAbstract: No

Generalization: MeasurableElement

Extension: Message

Description

Message for use in an Operational Event-Trace which carries any of the subtypes of OperationalExchange.

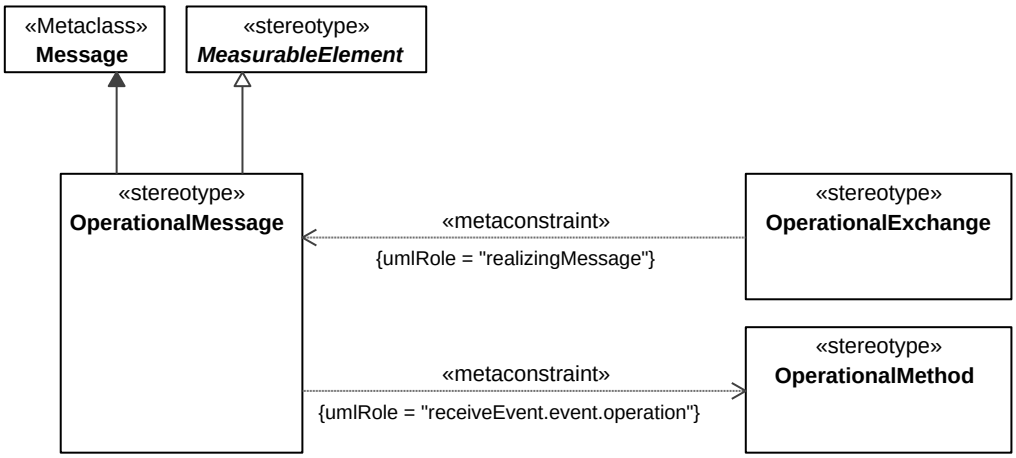

Figure 83 – OperationalMessage

Constraints

[1] OperationalMessage.receiveEvent.event.operation — Values for the receiveEvent.event.operation metaproperty must be stereotyped with «OperationalMethod» or its specializations.

7.1.5.7 UAF::Operational::Information

Contains the elements that contribute to the Operational Information Viewpoint.

InformationElement

Package: Information

isAbstract: No

Generalization: Asset, OperationalExchangeItem, SubjectOfOperationalConstraint

Extension: Class
Description

An item of information that flows between OperationalPerformers and is produced and consumed by the OperationalActivities that the OperationalPerformers are capable of performing (see IsCapableToPerform).

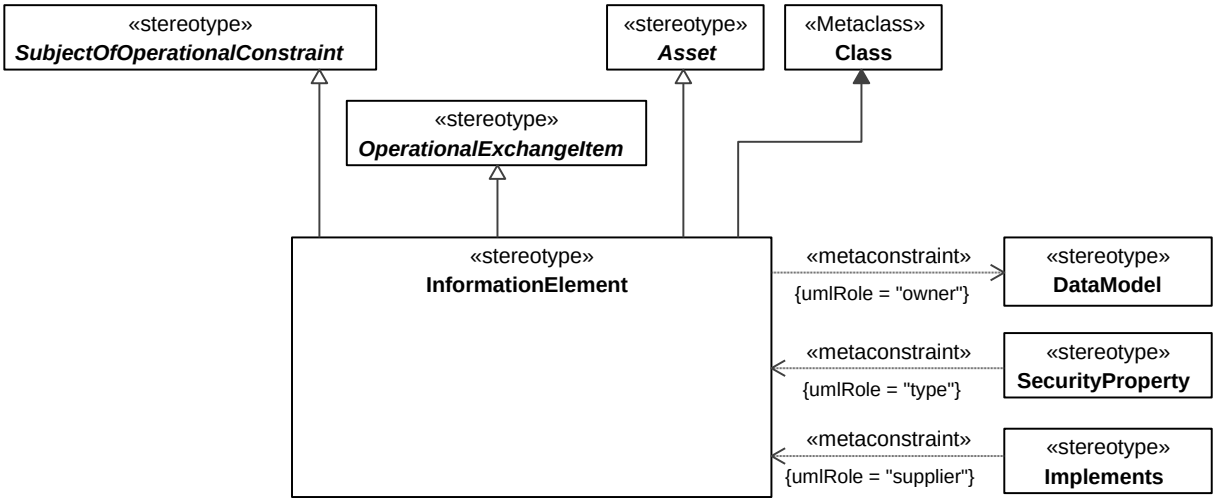

Figure 84 – InformationElement

Constraints

[1] InformationElement.owner — Values for the owner metaproperty must be stereotyped «DataModel» or its specializations.

7.1.5.8 UAF::Operational::Constraints

Contains the elements that contribute to the Operational Constraints Viewpoint.

OperationalConstraint

Package: Constraints

isAbstract: No

Generalization: Rule

Extension: Constraint
Description

A Rule governing a logical architectural element i.e. OperationalPerformer, OperationalActivity, InformationElement etc.

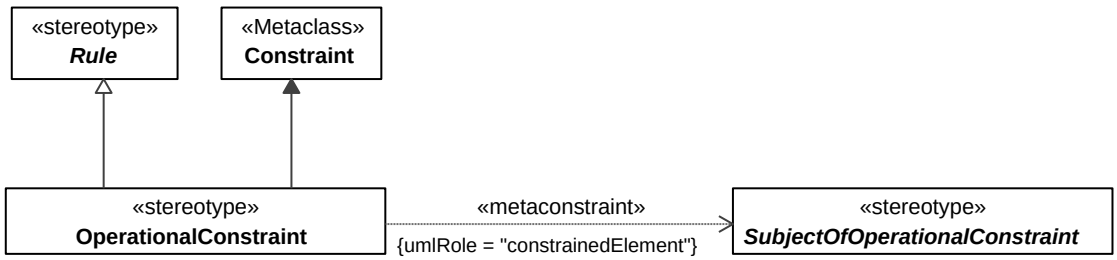

Figure 85 – OperationalConstraint

Constraints

[1] OperationalConstraint.constrainedElement — Value for the constrainedElement metaproperty must be stereotyped by any specialization of «SubjectOfOperationalConstraint».

SubjectOfOperationalConstraint

Package: Constraints

isAbstract: Yes

Generalization: UAFElement

Extension: Element
Description

An abstract grouping of elements that can be the subject of an OperationalConstraint.

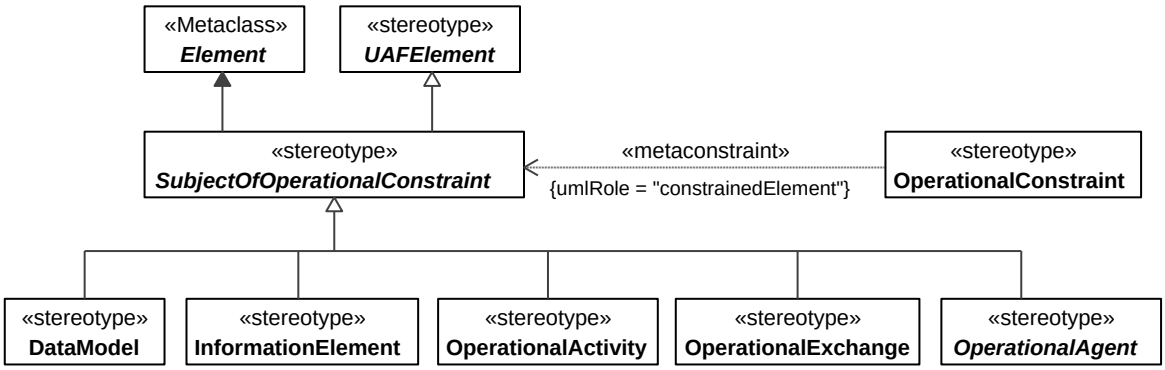

Figure 86 – SubjectOfOperationalConstraint

7.1.6 UAF::Services

Stakeholders: Enterprise Architects, Solution Providers, Systems Engineers, Software Architects, Business Architects..
Concerns: specifications of services required to exhibit a Capability.
Definition: shows Service Specifications and required and provided service levels of these specifications required to exhibit a Capability or to support an Operational Activity.

7.1.6.1 UAF::Services::Taxonomy

Contains the elements that contribute to the Services Taxonomy Viewpoint.

ServiceSpecification

Package: Taxonomy

isAbstract: No

Generalization: PropertySet, VersionedElement, CapableElement, Block

Extension: Class
Description

The specification of a set of functionality provided by one element for the use of others.

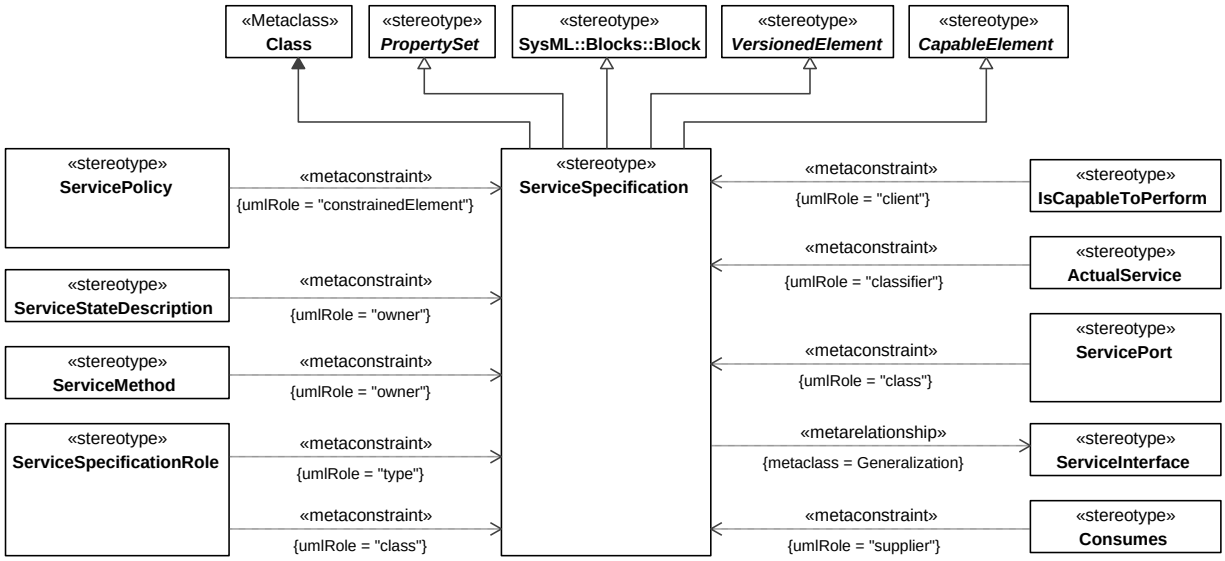

Figure 87 – ServiceSpecification

Constraints

[1] ServiceSpecification.isCapableToPerform — Is capable to perform only «ServiceFunction» elements or its specializations.

7.1.6.2 UAF::Services::Structure

Contains the elements that contribute to the Services Structure Viewpoint.

ServiceMethod

Package: Structure

isAbstract: No

Generalization: MeasurableElement

Extension: Operation

Description

A behavioral feature of a ServiceSpecification whose behavior is specified in a ServiceFunction.

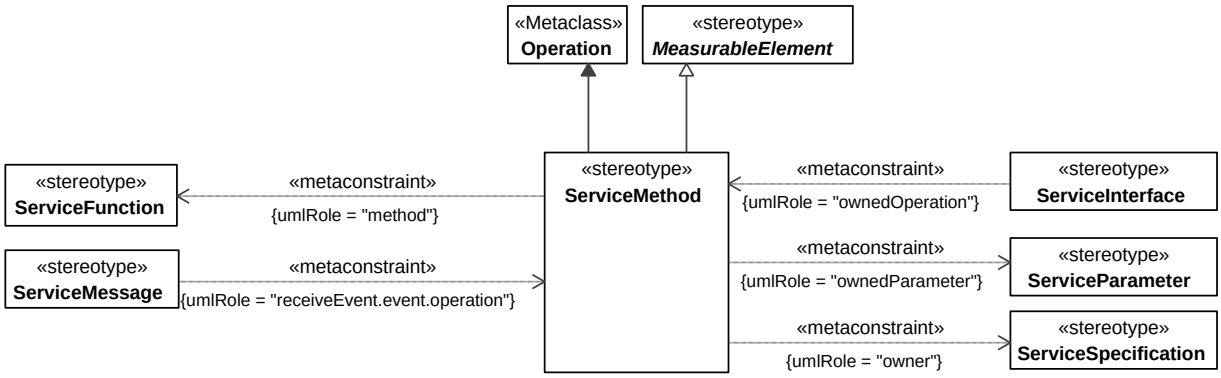

Figure 88 – ServiceMethod

Constraints

[1] ServiceMethod.method — Value for the method metaproperty must be stereotyped «ServiceFunction» or its specializations.

[2] ServiceMethod.ownedParameter — The values for the ownedParameter metaproperty must be stereotyped «ServiceParameter» or its specializations.

[3] ServiceMethod.owner	The values for the owner metaproperty must be stereotyped «ServiceSpecification» or its specializations.

ServiceParameter

Package: Structure

isAbstract: No

Generalization: MeasurableElement

Extension: Parameter

Description

An element that represents inputs and outputs of a ServiceFunction, represents inputs and outputs of a ServiceSpecification.

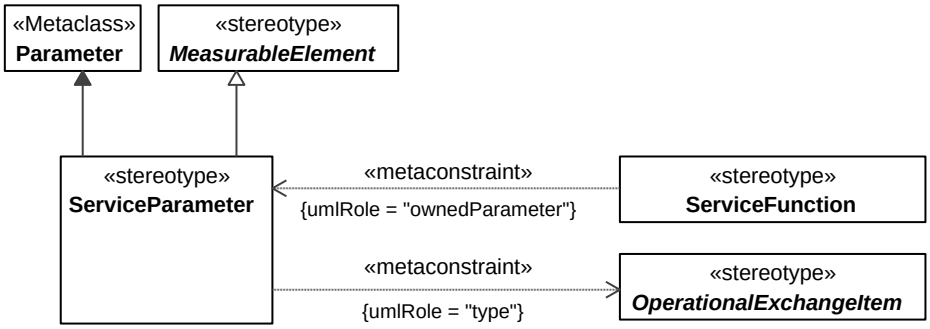

Figure 89 – ServiceParameter

Constraints

[1] ServiceParameter.type	The values for the type metaproperty must be stereotyped a specialization of «OperationalExchangeItem».

ServicePort

Package: Structure

isAbstract: No

Generalization: ProxyPort, MeasurableElement

Extension: Port

Description

An interaction point for a ServiceSpecification through which it can interact with the outside environment and which is defined by a ServiceInterface.

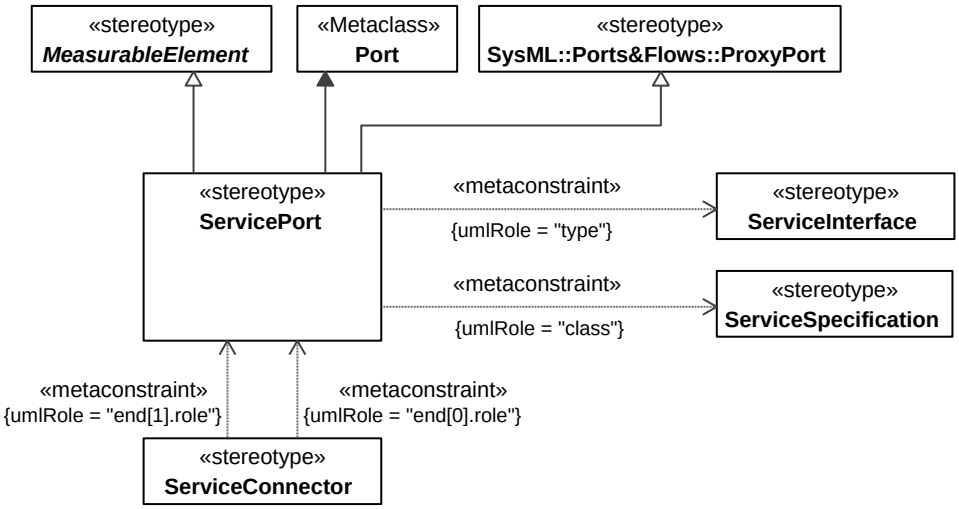

Figure 90 – ServicePort

Constraints

[1] ServicePort.class Value for the class metaproperty must be stereotyped «ServiceSpecification» or its specializations.

[2] ServicePort.type Value for the type metaproperty must be stereotyped «ServiceInterface» or its specializations.

ServiceSpecificationRole

Package: Structure

isAbstract: No

Generalization: MeasurableElement

Extension: Property

Description

Usage of a ServiceSpecification in the context of another ServiceSpecification. Creates a whole-part relationship.

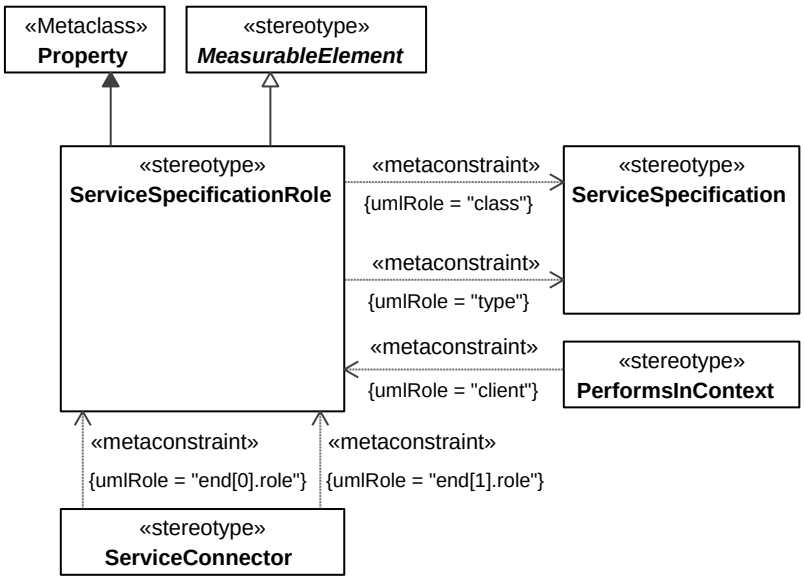

Figure 91 – ServiceSpecificationRole

Constraints

[1] ServiceSpecificationRole.class Value for the class metaproperty must be stereotyped «ServiceSpecification» or its

specializations.

[2] ServiceSpecificationRole.type	Value for the type metaproperty must be stereotyped «ServiceSpecification» or its specializations.

7.1.6.3 UAF::Services::Connectivity

Contains the elements that contribute to the Services Connectivity Viewpoint.

ServiceConnector

Package: Connectivity

isAbstract: No

Generalization: MeasurableElement

Extension: Connector

Description

A channel for exchange between two ServiceSpecifications. Where one acts as the consumer of the other.

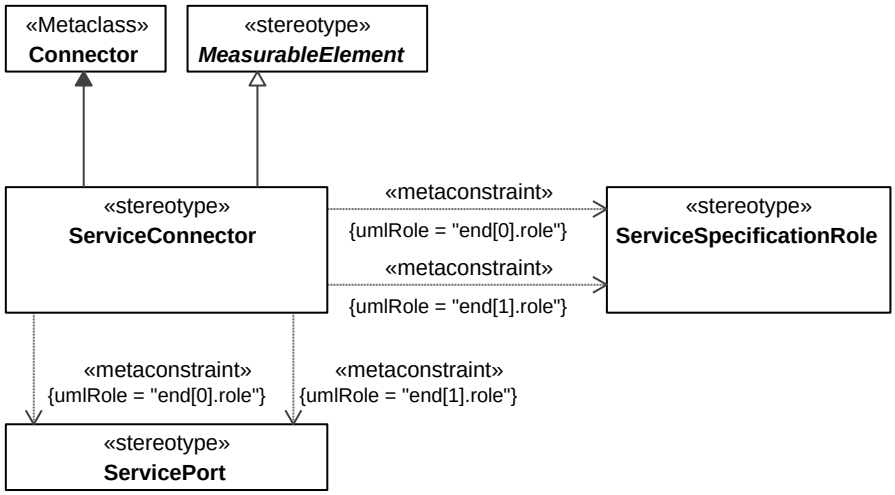

Figure 92 – ServiceConnector

Constraints

[1] ServiceConnector.end	The value for the role metaproperty for the owned ConnectorEnd must be stereotyped «ServicePort», «ServiceSpecificationRole» or their specializations.

ServiceInterface

Package: Connectivity

isAbstract: No

Generalization: PropertySet, InterfaceBlock

Extension: Class

Description

A contract that defines the ServiceMethods and ServiceMessageHandlers that the ServiceSpecification realizes.

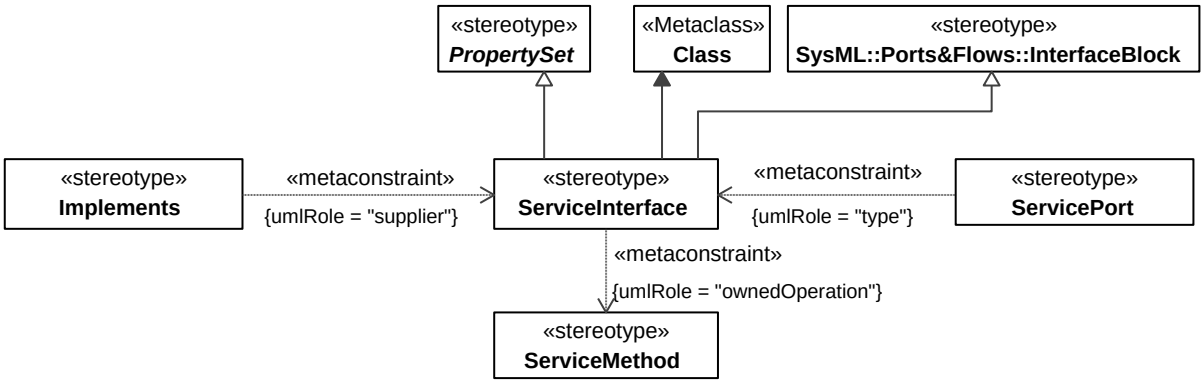

Figure 93 – ServiceInterface

Constraints

[1] ServiceInterface.ownedOperation — Values for the ownedOperation metaproperty must be stereotyped «ServiceMethod» or its specializations.

7.1.6.4 UAF::Services::Processes

Contains the elements that contribute to the Services Processes Viewpoint.

ServiceFunction

Package: Processes

isAbstract: No

Generalization: Activity

Extension: Activity

Description

An Activity that describes the abstract behavior of ServiceSpecifications, regardless of the actual implementation.

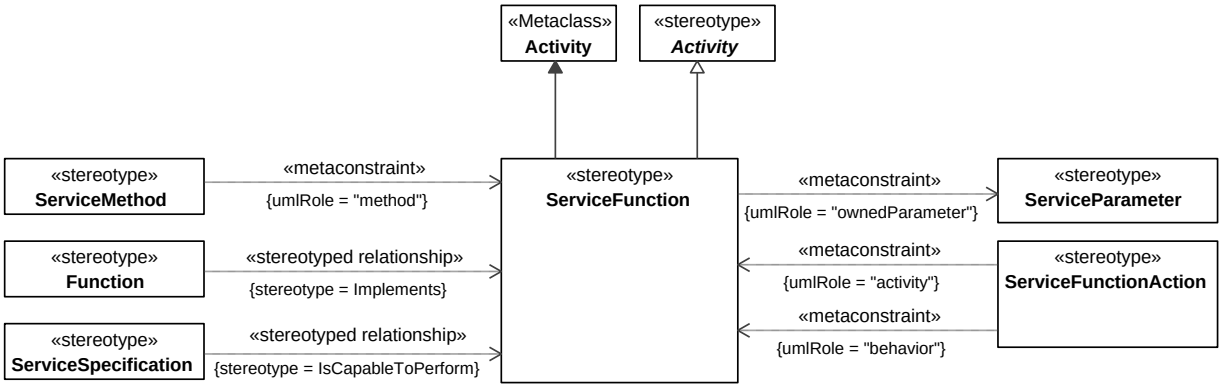

Figure 94 – ServiceFunction

Constraints

[1] ServiceFunction.ownedParameter — The values for the ownedParameter metaproperty must be stereotyped «ServiceParameter».

ServiceFunctionAction

Package: Processes

isAbstract: No

Generalization: MeasurableElement

Extension: CallBehaviorAction
Description

A call of a ServiceFunction in the context of another ServiceFunction.

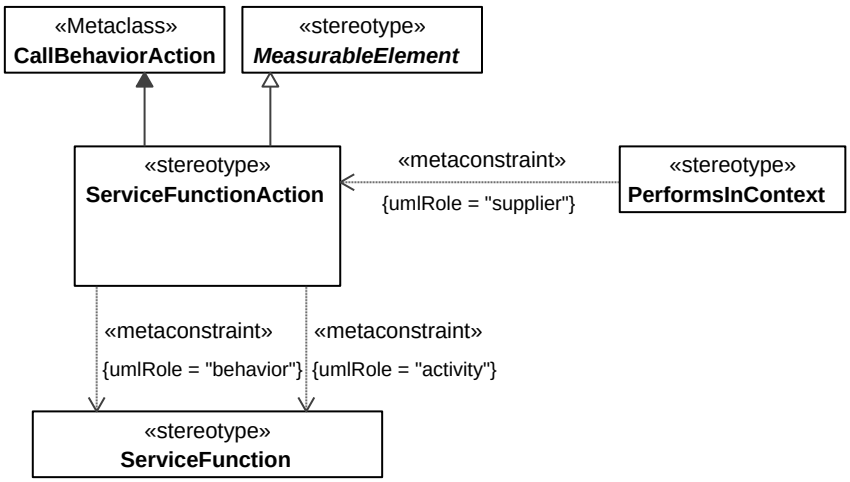

Figure 95 – ServiceFunctionAction

Constraints

[1] ServiceFunctionAction.activity	Value for the behavior metaproperty must be stereotyped «ServiceFunction» or its specializations.
[2] ServiceFunctionAction.behavior	Value for the activity metaproperty must be stereotyped «ServiceFunction» or its specializations.

7.1.6.5 UAF::Services::States

Contains the elements that contribute to the Services States Viewpoint.

ServiceStateDescription

Package: States

isAbstract: No

Generalization: MeasurableElement

Extension: StateMachine
Description

A state machine describing the behavior of a ServiceSpecification, depicting how the ServiceSpecification responds to various events and the actions.

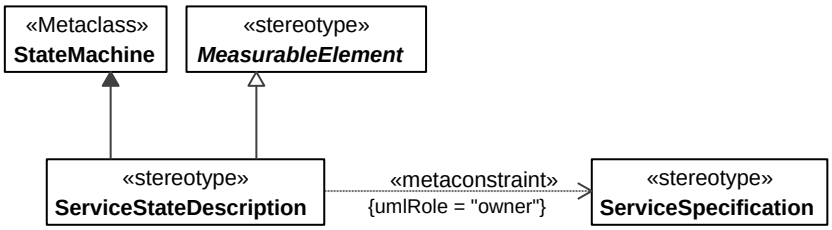

Figure 96 – ServiceStateDescription

Constraints

[1] ServiceStateMachine.owner	Values for the owner metaproperty must be stereotyped «ServiceSpecification» or its specializations.

7.1.6.6 UAF::Services::Interaction Scenarios

Contains the elements that contribute to the Services Interaction Scenarios Viewpoint.

ServiceMessage

Package: Interaction Scenarios

isAbstract: No

Generalization: MeasurableElement

Extension: Message
Description

Message for use in a Service Event-Trace.

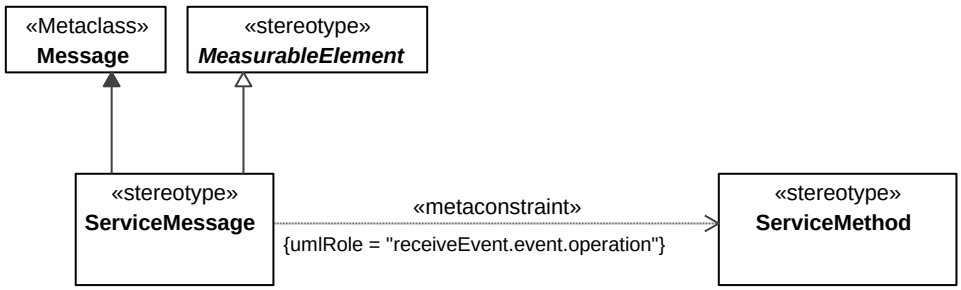

Figure 97 – ServiceMessage

Constraints

[1] ServiceMessage.receiveEvent.event.operation — Values for the receiveEvent.event.operation metaproperty must be stereotyped with «ServiceMethod» or its specializations.

7.1.6.7 UAF::Services::Constraints

Contains the elements that contribute to the Services Constraints Viewpoint.

ServicePolicy

Package: Constraints

isAbstract: No

Generalization: Rule

Extension: Constraint
Description

A constraint governing the use of one or more ServiceSpecifications.

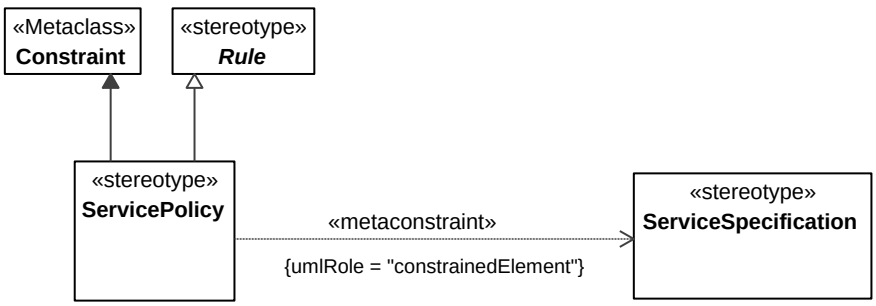

Figure 98 – ServicePolicy

Constraints

[1] ServicePolicy.constrainedElement — Values for constrainedElement metaproperty must be stereotyped «ServiceSpecification» or its specializations.

7.1.6.8 UAF::Services::Traceability

Contains the elements that contribute to the Services Traceability Viewpoint.

Consumes

Package: Traceability

isAbstract: No

Generalization: Allocate, MeasurableElement

Extension: Abstraction

Description

A abstraction relationship that asserts that a service in someway contributes or assists in the execution of an OperationalActivity.

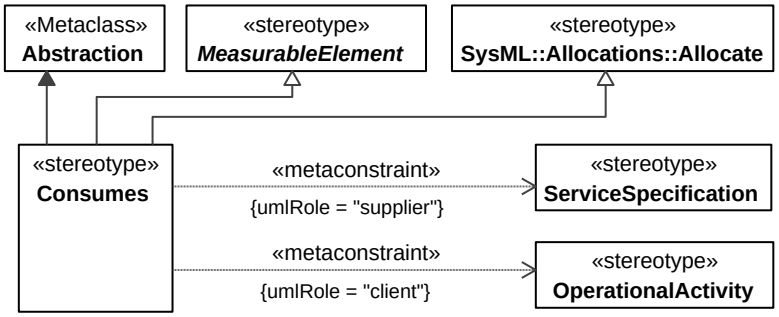

Figure 99 – Consumes

Constraints

[1] Consumes.client	Value for the client metaproperty must be stereotyped «OperationalActivity» or its specializations.
[2] Consumes.supplier	Value for the supplier metaproperty must be stereotyped «ServiceSpecification» or its specializations.

7.1.7 UAF::Personnel

Stakeholders: Human resources, Solution Providers, PMs.
Concerns: human factors.
Definition: aims to clarify the role of Human Factors (HF) when creating architectures in order to facilitate both Human Factors Integration (HFI) and systems engineering (SE).

7.1.7.1 UAF::Personnel::Taxonomy

Contains the elements that contribute to the Personnel Taxonomy Viewpoint.

Organization

Package: Taxonomy

isAbstract: No

Generalization: OrganizationalResource

Extension: Class

Description

A group of OrganizationalResources (Persons, Posts, Organizations and Responsibilities) associated for a particular purpose.

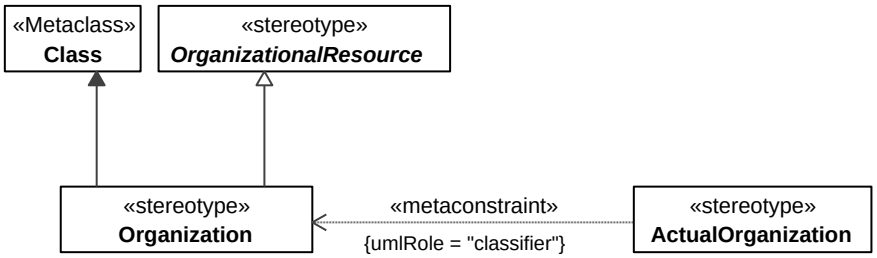

Figure 100 – Organization

OrganizationalResource

Package: Taxonomy

isAbstract: Yes

Generalization: PhysicalResource, Stakeholder

Extension: Class
Description

An abstract element grouping for Organization, Person Post and Responsibility.

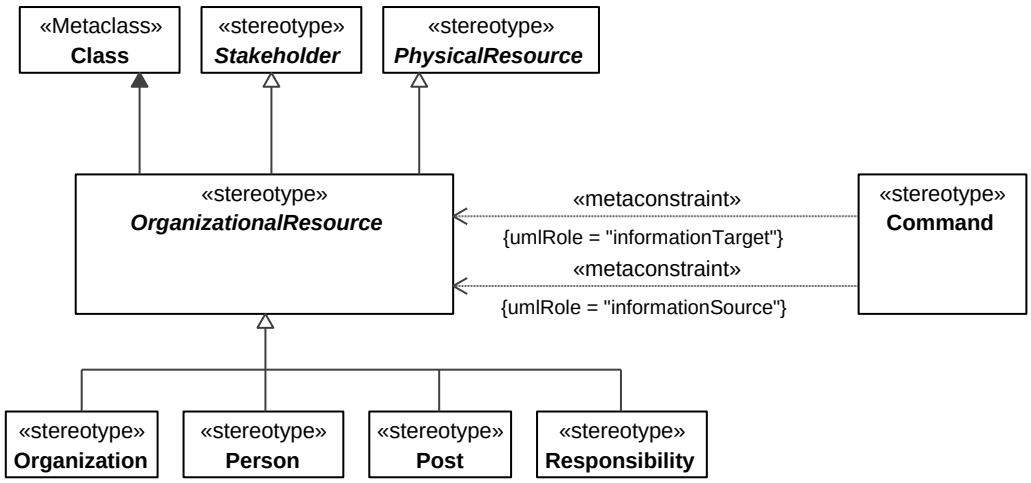

Figure 101 – OrganizationalResource

Person

Package: Taxonomy

isAbstract: No

Generalization: OrganizationalResource

Extension: Class
Description

A type of a human being used to define the characteristics that need to be described for ActualPersons (e.g. properties such as address, telephone number, nationality, etc).

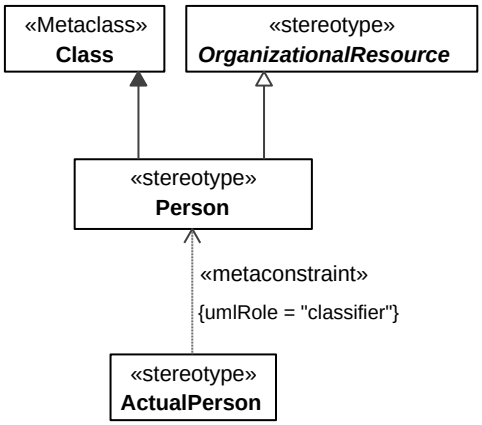

Figure 102 – Person

Post

Package: Taxonomy

isAbstract: No

Generalization: OrganizationalResource

Extension: Class
Description

A type of job title or position that a person can fill (e.g. Lawyer, Solution Architect, Machine Operator or Chief Executive Officer).

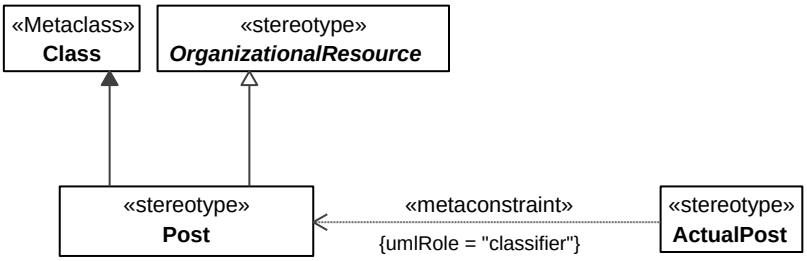

Figure 103 – Post

Responsibility

Package: Taxonomy

isAbstract: No

Generalization: OrganizationalResource

Extension: Class
Description

The type of duty required of a Person or Organization.

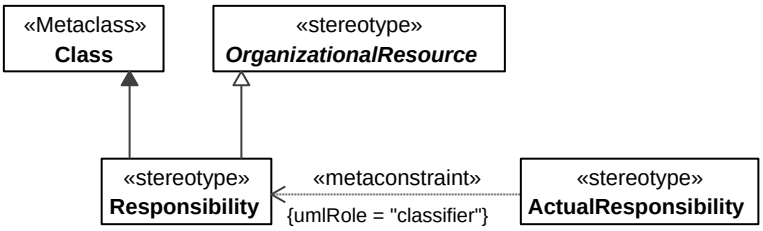

Figure 104 – Responsibility

7.1.7.2 UAF::Personnel::Connectivity

Contains the elements that contribute to the Personnel Connectivity Viewpoint.

Command

Package: Connectivity

isAbstract: No

Generalization: ResourceExchange

Extension: InformationFlow

Description

A type of ResourceExchange that asserts that one OrganizationalResource commands another.

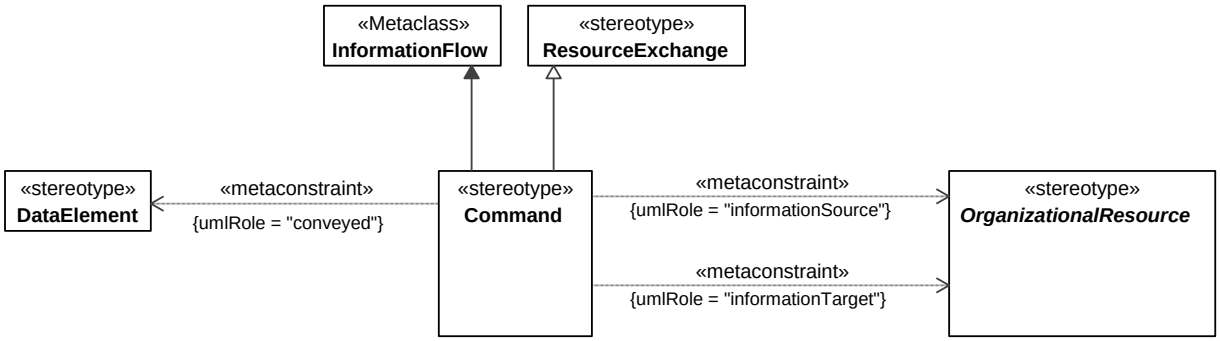

Figure 105 – Command

Constraints

[1] Command.conveyed	Value for the conveyed metaproperty must be stereotyped «DataElement» or its specializations.
[2] Command.informationSource	Value for the informationSource metaproperty must be stereotyped by the specialization of «OrganizationalResource».
[3] Command.informationTarget	Value for the informationTarget metaproperty must be stereotyped by the specialization of «OrganizationalResource».

Control

Package: Connectivity

isAbstract: No

Generalization: ResourceExchange

Extension: InformationFlow

Description

A type of ResourceExchange that asserts that one PhysicalResource controls another PhysicalResource (i.e. the driver of a vehicle controlling the vehicle speed or direction).

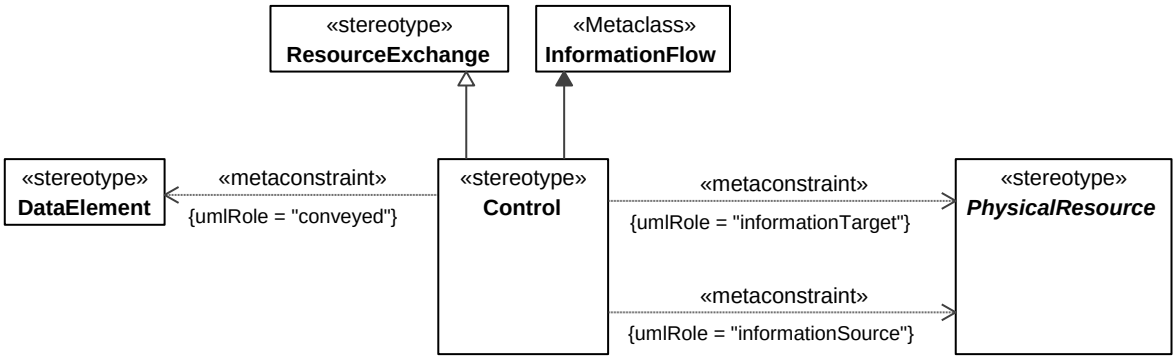

Figure 106 – Control

Constraints

[1] Control.conveyed	Value for the conveyed metaproperty must be stereotyped «DataElement» or its specializations.
[2] Control.informationSource	Value for the informationSource metaproperty must be stereotyped by the specialization of «PhysicalResource».
[3] Control.informationTarget	Value for the informationTarget metaproperty must be stereotyped by the specialization of «PhysicalResource» or its specializations.

7.1.7.3 UAF::Personnel::Processes

Contains the elements that contribute to the Personnel Processes Viewpoint.

CompetenceToConduct

Package: Processes

isAbstract: No

Generalization: MeasurableElement, Allocate

Extension: Abstraction

Description

An abstraction relationship used to associate a Function with a specific set of Competencies needed to conduct the Function.

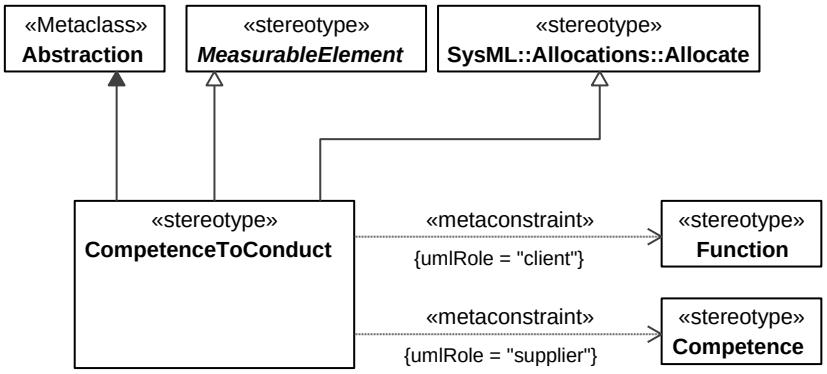

Figure 107 – CompetenceToConduct

Constraints

[1] CompetenceToConduct.client	Value for the client metaproperty must be stereotyped «Function» or its specializations.
[2] CompetenceToConduct.supplier	Value for the supplier metaproperty must be stereotyped «Competence» or its specializations.

7.1.7.4 UAF::Personnel::Constraints

Contains the elements that contribute to the Personnel Constraints Viewpoint.

Competence

Package: Constraints

isAbstract: No

Generalization: SubjectOfForecast, PropertySet, Block

Extension: Class
Description

A specific set of abilities defined by knowledge, skills and aptitude.

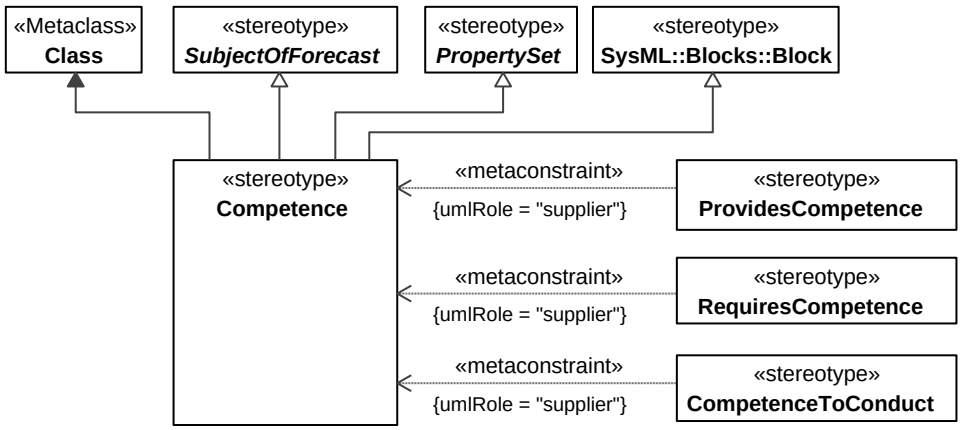

Figure 108 – Competence

CompetenceForRole

Package: Constraints

isAbstract: No

Generalization: MeasurableElement, Allocate

Extension: Abstraction
Description

An abstraction relationship used to associate an organizational role with a specific set of required competencies.

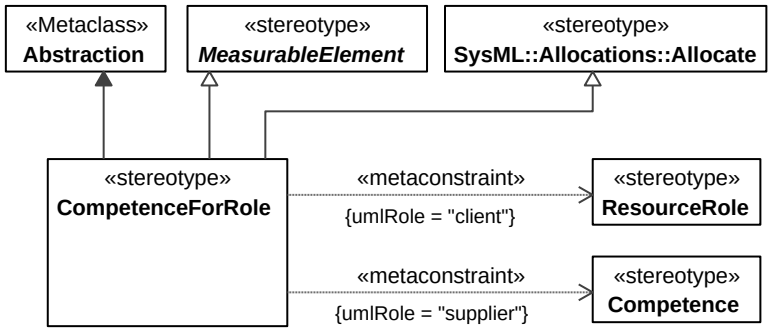

Figure 109 – CompetenceForRole

Constraints

[1] CompetenceForRole.client	Value for the client metaproperty must be stereotyped «ResourceRole» or its specializations.
[2] CompetenceForRole.supplier	Value for the supplier metaproperty must be stereotyped «Competence» or its

specializations.

RequiresCompetence

Package: Constraints

isAbstract: No

Generalization: MeasurableElement, Allocate

Extension: Abstraction

Description

An abstraction relationship that asserts that an ActualOrganizationalResource is required to have a specific set of Competencies.

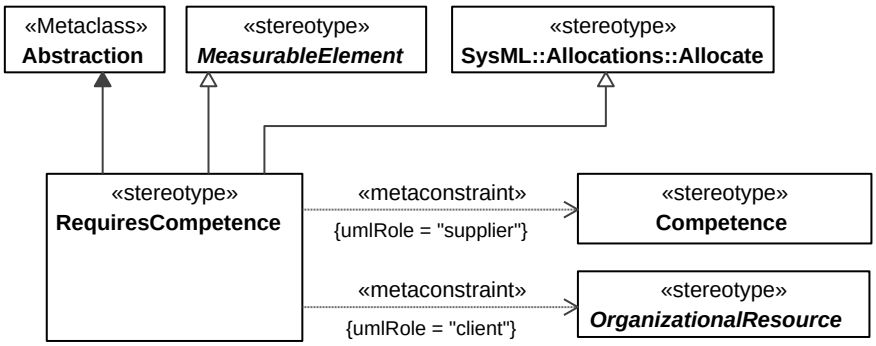

Figure 110 – RequiresCompetence

Constraints

[1] RequiresCompetence.client	Value for the client metaproperty must be stereotyped a specialization of «OrganizationalResource».
[2] RequiresCompetence.supplier	Value for the supplier metaproperty must be stereotyped «Competence» or its specializations.

7.1.7.5 UAF::Personnel::Traceability

Contains the elements that contribute to the Personnel Traceability Viewpoint.

ResponsibleFor

Package: Traceability

isAbstract: No

Generalization: MeasurableElement, Allocate

Extension: Abstraction

Description

An abstraction relationship between an ActualResponisbleResource and an ActualResponsibility or ActualProject. It defines the duties that the ActualResponsibleResource is ResponisbleFor.

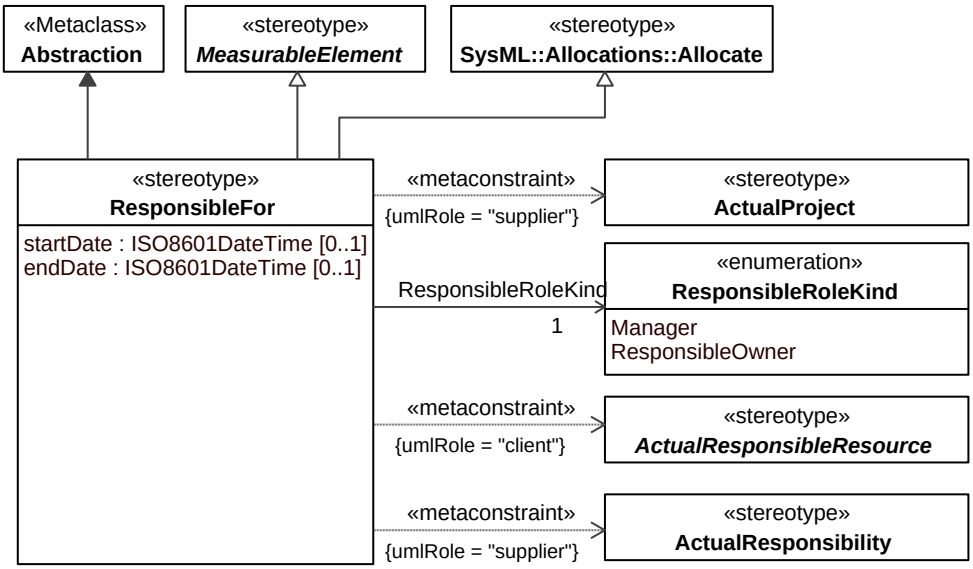

Figure 111 – ResponsibleFor

Attributes

endDate : ISO8601DateTime[0..1] End date of an ActualResponsibleResource being ResponsibleFor and ActualProject or ActualResponsibility.

startDate : ISO8601DateTime[0..1] Start date of an ActualResponsibleResource being ResponsibleFor and ActualProject or ActualResponsibility.

Associations

ResponsibleRoleKind : ResponsibleRoleKind[1] Captures the kind of role (Manager or ResponsibleOwner) responsible for the ActualProject or ActualResponsibility.

Constraints

[1] ResponsibleFor.client Value for the client metaproperty must be stereotyped by the specialization of «ActualResponsibleResource».

[2] ResponsibleFor.supplier Value for the supplier metaproperty must be stereotyped «ActualProject», «ActualResponsibility», or their specializations.

ResponsibleRoleKind

Package: Traceability

isAbstract: No
Description

Enumeration of the possible kinds or ResponsibleRole. Its enumeration literals are:

- Manager - Indicates that the ResourceInteraction associated with the ResourceInteractionKind is a an implementation of logical flow.
- ResponsibleOwner - Indicates that the ResourceInteraction associated with the ResourceInteractionKind is a an implementation of logical flow.

7.1.8 UAF::Resources

Stakeholders: Systems Engineers, Resource Owners, Implementers, Solution Providers, IT Architects.
Concerns: definition of solution architectures to implement operational requirements. Definition: captures a solution architecture consisting of resources, e.g. organizational, software, artifacts, capability configurations, natural resources that implement the operational requirements. Further design of a resource is typically detailed in SysML or UML.

7.1.8.1 UAF::Resources::Taxonomy

Contains the elements that contribute to the Resources Taxonomy Viewpoint.

CapabilityConfiguration

Package: Taxonomy

isAbstract: No

Generalization: ResourceArchitecture

Extension: Class
Description

A composite structure representing the physical and human resources (and their interactions) in an enterprise, assembled to meet a capability.

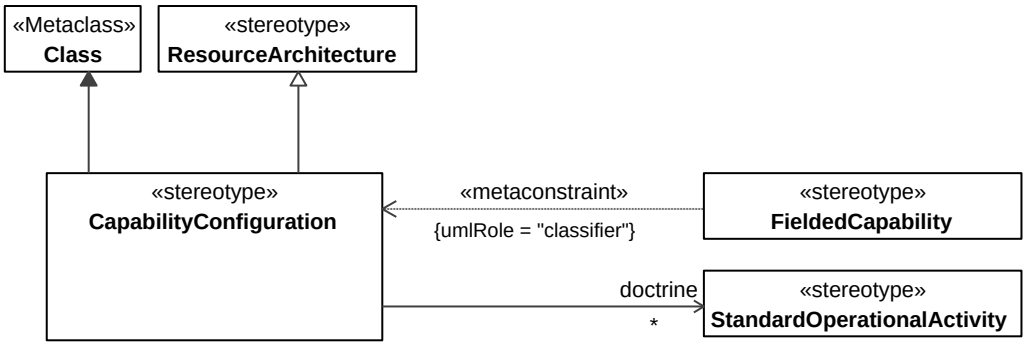

Figure 112 – CapabilityConfiguration

Associations

doctrine : StandardOperationalActivity[*] Represents the doctrinal line of development of the Capability.

NaturalResource

Package: Taxonomy

isAbstract: No

Generalization: PhysicalResource

Extension: Class
Description

Type of physical resource that occurs in nature such as oil, water, gas or coal.

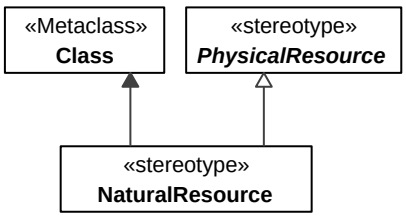

Figure 113 – NaturalResource

PhysicalResource

Package: Taxonomy

isAbstract: Yes

Generalization: ResourcePerformer

Extension: Class
Description

An abstract grouping that defines physical resources (i.e. OrganizationalResource, ResourceArtifact and NaturalResource).

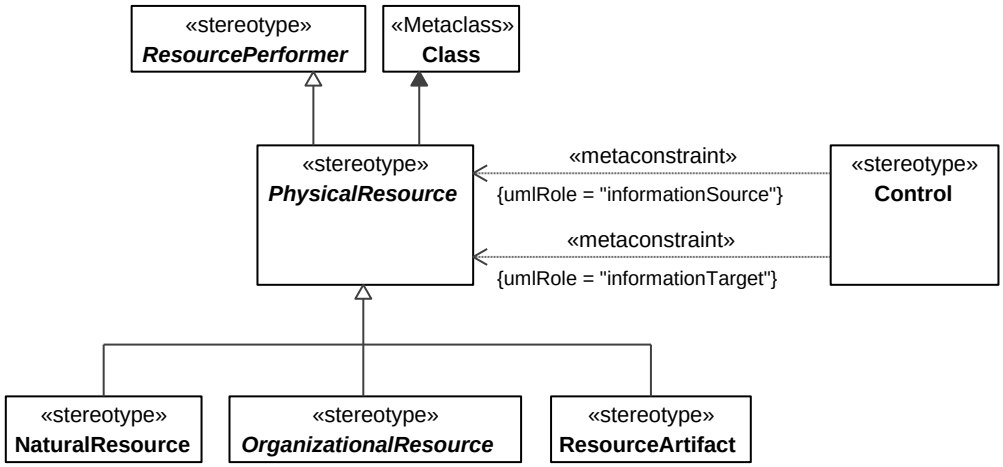

Figure 114 – PhysicalResource

ResourceArchitecture

Package: Taxonomy

isAbstract: No

Generalization: ResourcePerformer, Architecture

Extension: Class
Description

An element used to denote a model of the Architecture, described from the ResourcePerformer perspective.

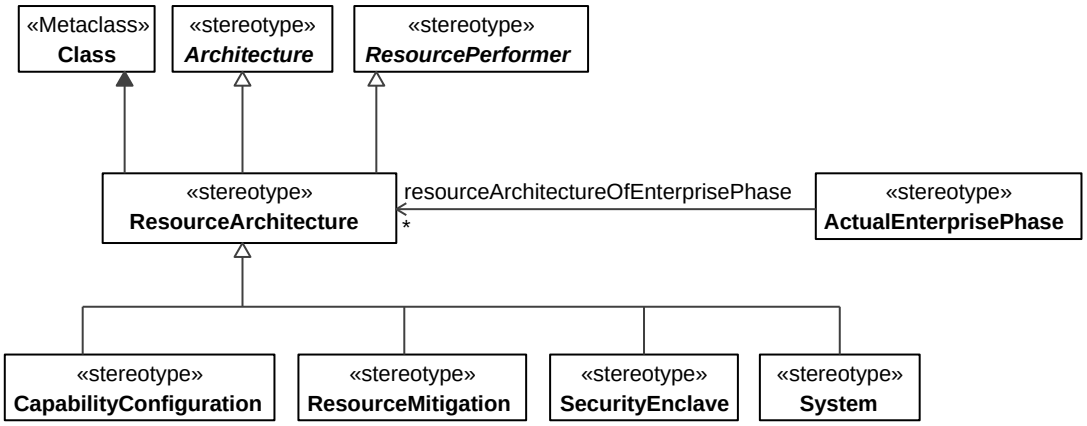

Figure 115 – ResourceArchitecture

ResourceArtifact

Package: Taxonomy

isAbstract: No

Generalization: PhysicalResource

Extension: Class
Description

A type of man-made object that contains no human beings (i.e. satellite, radio, petrol, gasoline, etc.).

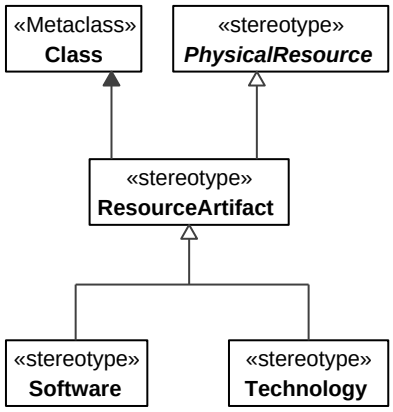

Figure 116 – ResourceArtifact

ResourcePerformer

Package: Taxonomy

isAbstract: Yes

Generalization: Asset, ResourceExchangeItem, SubjectOfResourceConstraint, VersionedElement, CapableElement, SubjectOfForecast, OperationalExchangeItem, Desirer

Extension: Class
Description

An abstract grouping of elements that can perform Functions.

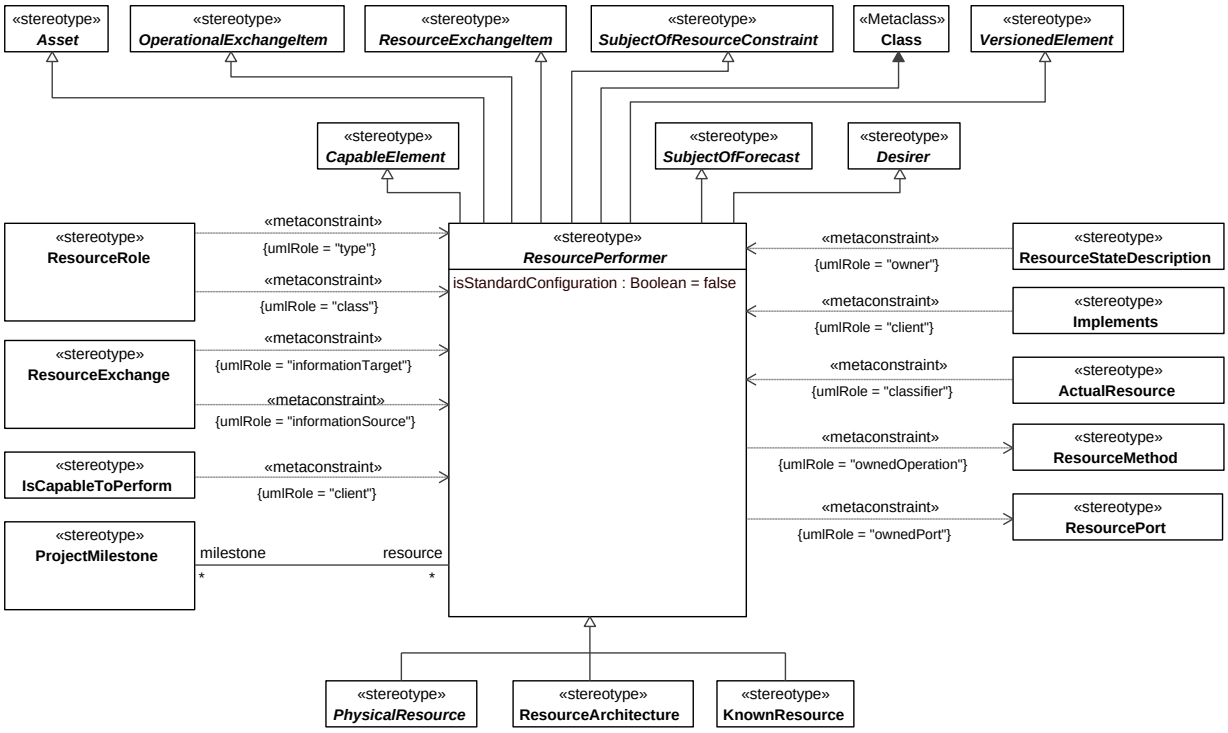

Figure 117 – ResourcePerformer

Attributes

isStandardConfiguration : Boolean[] Indicates if the ResourcePerformer is StandardConfiguration, default=false.
Associations

milestone : ProjectMilestone[*] Relates ResourcePerformer to ProjectMilestones that affect it.

Constraints

[1] ResourcePerformer.isCapableToPerform	Is capable to perform only «Function» elements or its specializations.
[2] ResourcePerformer.ownedOperation	Values for the ownedOperation metaproperty must be stereotyped «ResourceMethod» or its specializations.
[3] ResourcePerformer.ownedPort	Values for the ownedPort metaproperty must be stereotyped «ResourcePort» or its specializations.

Software

Package: Taxonomy

isAbstract: No

Generalization: ResourceArtifact

Extension: Class
Description

A sub-type of ResourceArtifact that specifies an executable computer program.

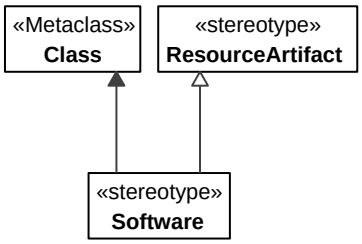

Figure 118 – Software

System

Package: Taxonomy

isAbstract: No

Generalization: ResourceArchitecture

Extension: Class
Description

An integrated set of elements, subsystems, or assemblies that accomplish a defined objective. These elements include products (hardware, software, firmware), processes, people, information, techniques, facilities, services, and other support elements (INCOSE SE Handbook V4, 2015).

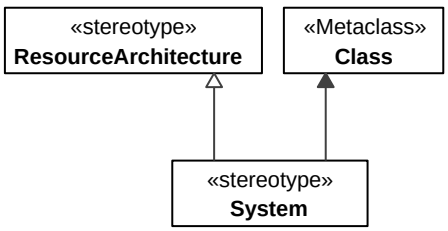

Figure 119 – System

7.1.8.2 UAF::Resources::Structure

Contains the elements that contribute to the Resources Structure Viewpoint.

ResourceMethod

Package: Structure

isAbstract: No

Generalization: MeasurableElement

Extension: Operation

Description

A behavioral feature of a ResourcePerformer whose behavior is specified in a Function.

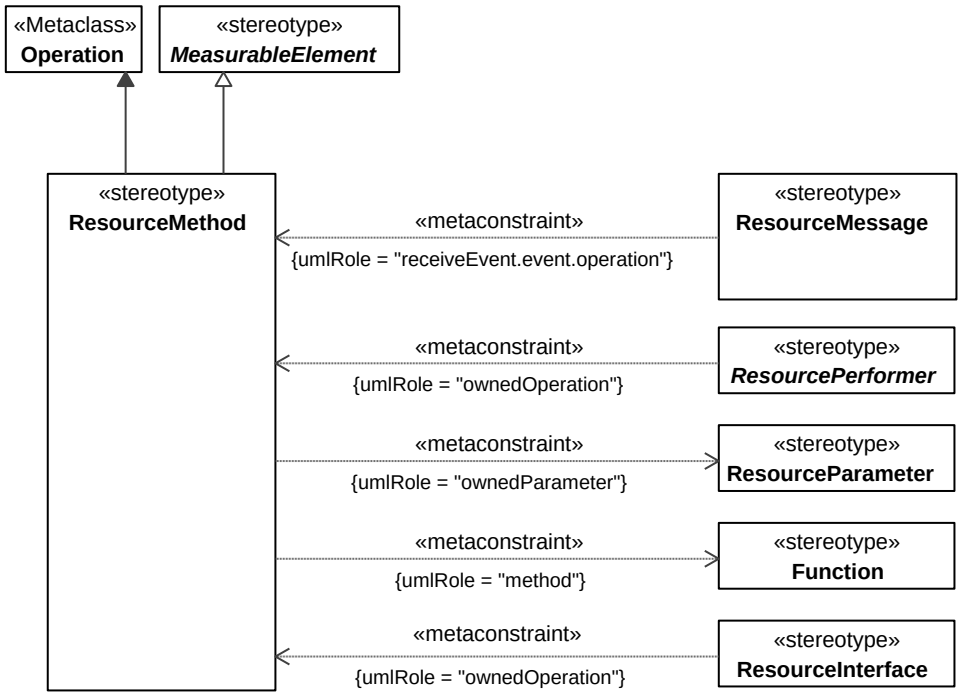

Figure 120 – ResourceMethod

Constraints

[1] ResourceMethod.method	Value for the method metaproperty must be stereotyped «Function» or its specializations.
[2] ResourceMethod.ownedParameter	The values for the ownedParameter metaproperty must be stereotyped «ResourceParameter».

ResourceParameter

Package: Structure

isAbstract: No

Generalization: MeasurableElement

Extension: Parameter

Description

An element that represents inputs and outputs of an Function. It is typed by a ResourceInteractionItem.

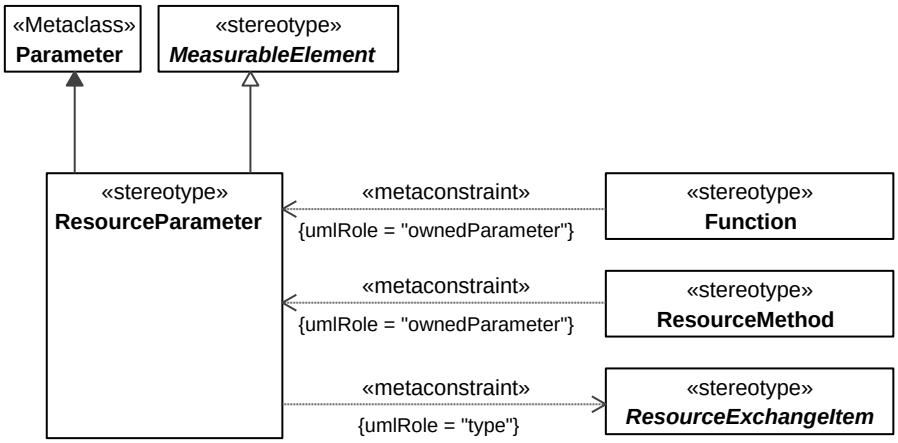

Figure 121 – ResourceParameter

Constraints

[1] ResourceParameter.type Value for the type metaproperty must be stereotyped with a specialization of «ResourceInteractionItem».

ResourcePort

Package: Structure

isAbstract: No

Generalization: ProxyPort, MeasurableElement, ProtocolImplementation

Extension: Port

Description

An interaction point for a ResourcePerformer through which it can interact with the outside environment and which is defined by a ResourceInterface.

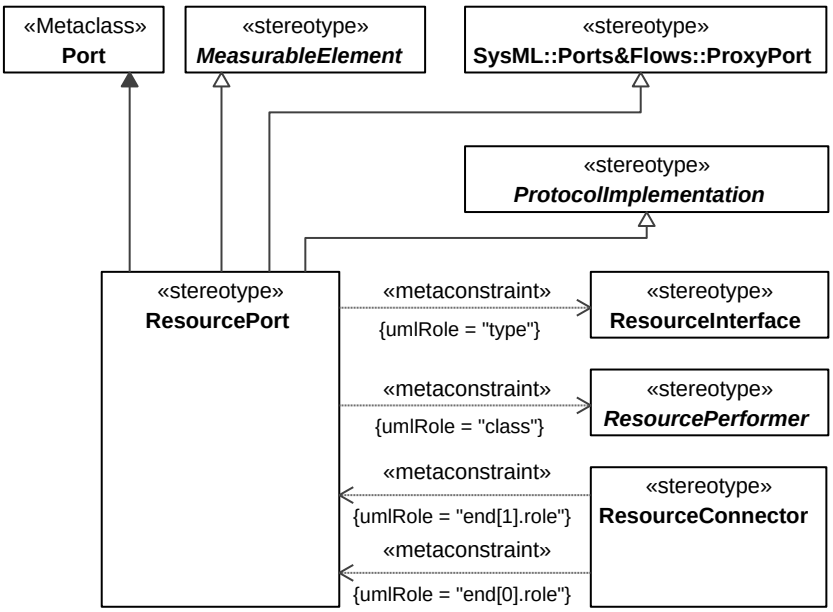

Figure 122 – ResourcePort

Constraints

[1] ResoucePort.type Value for the type metaproperty must be stereotyped «ResourceInterface» or its specializations.

[2] ResourcePort.class Value for the class metaproperty must be stereotyped by the specialization of

«ResourcePerformer».

ResourceRole

Package: Structure

isAbstract: No

Generalization: LocationHolder, SubjectOfResourceConstraint, MeasurableElement, SubjectOfSecurityConstraint, AssetRole

Extension: Property
Description

Usage of a ResourcePerformer in the context of another ResourcePerformer. Creates a whole-part relationship.

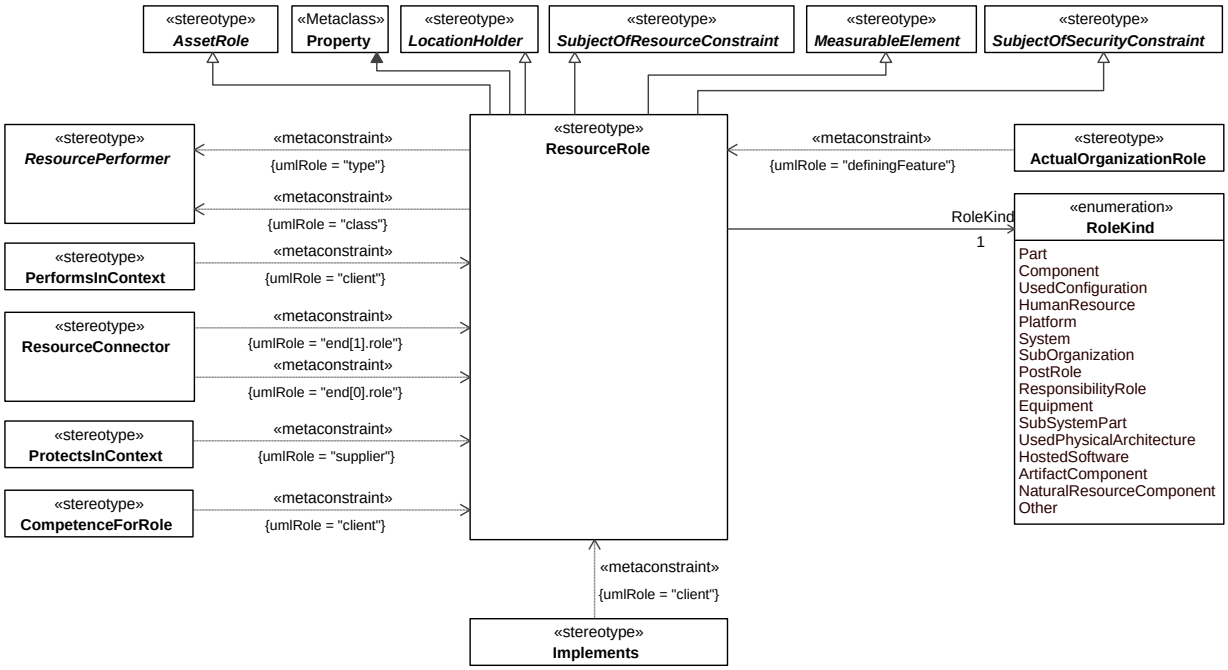

Figure 123 – ResourceRole

Associations

RoleKind : RoleKind[1] Captures the kind of role a Resource can play.

Constraints

[1] ResouceRole.type Value for the type metaproperty must be stereotyped by the specialization of «ResourcePerformer».

[2] ResourceRole.class Value for the class metaproperty must be stereotyped by the specialization of «ResourcePerformer».

RoleKind

Package: Structure

isAbstract: No
Description

Enumeration of the possible kinds of roles that a ResourceRole may play in the context of a ResourcePerformer. Its enumeration literals are:

- Part - Indicates that the ResourceRole associated with the ResourceRoleKind is a kind of a ResourcePerformer that is used as a part of another ResourcePerformer.
- Component - Indicates that the ResourceRole associated with the ResourceRoleKind is a kind of Software that is used in the context of a ResourcePerformer.

- UsedConfiguration - Indicates that the ResourceRole associated with the ResourceRoleKind is a kind of existing CapabilityConfiguration that is used in the context of a ResourcePerformer.
- HumanResource - Indicates that the ResourceRole associated with the ResourceRoleKind is a kind of human resource that is used in the context of a ResourcePerformer.
- Platform - Indicates that the ResourceRole associated with the ResourceRoleKind is a kind of a ResourcePerformer that represents a platform (e.g. vessel, aircraft, etc.) that is used in the context of a SystemsResource.
- System - Indicates that the ResourceRole associated with the ResourceRoleKind is a kind of assembly of ResourcePerformers that is used in the context of another ResourcePerformer.
- SubOrganization - Indicates that the ResourceRole associated with the ResourceRoleKind is a kind of Organization that is typically the parent of another - e.g. a squadron may be part of a batallion, that is used in the context of a ResourcePerformer.
- PostRole - Indicates that the ResourceRole associated with the ResourceRoleKind is a kind of Post that is used in the context of a ResourcePerformer.
- ResponsibilityRole - Indicates that the ResourceRole associated with the ResourceRoleKind is a kind of Responsibility associated with a role that is used in the context of a ResourcePerformer.
- Equipment - Indicates that the ResourceRole associated with the ResourceRoleKind is a kind of man made resource that is used to accomplish a task or function in the context of a ResourcePerformer.
- SubSystemPart - Indicates that the ResourceRole associated with the ResourceRoleKind is a kind of subsystem (represented as a ResourcePerformers) is is part of another ResourcePerformer.
- UsedPhysicalArchitecture - Indicates that the ResourceRole associated with the ResourceRoleKind is a kind of existing PhysicalArchitecture that is used in the context of a ResourcePerformer.
- HostedSoftware - Indicates that the ResourceRole associated with the ResourceRoleKind is a kind of software that is used in the context of a ResourcePerformer.
- ArtifactComponent - Indicates that the ResourceRole associated with the ResourceRoleKind is a kind of non human resource that is used as a component in the context of a ResourcePerformer.
- NaturalResourceComponent - Indicates that the ResourceRole associated with the ResourceRoleKind is a kind of natural resource that is used as a component in the context of a ResourcePerformer.
- Other - Indicates that the ResourceRole associated with the ResourceRoleKind is another kind of RoleKind that is not on the enumerated list.

7.1.8.3 UAF::Resources::Connectivity

Contains the elements that contribute to the Resources Connectivity Viewpoint.

ResourceConnector

Package: Connectivity

isAbstract: No

Generalization: MeasurableElement, ProtocolImplementation

Extension: Connector

Description

A channel for exchange between two ResourceRoles.

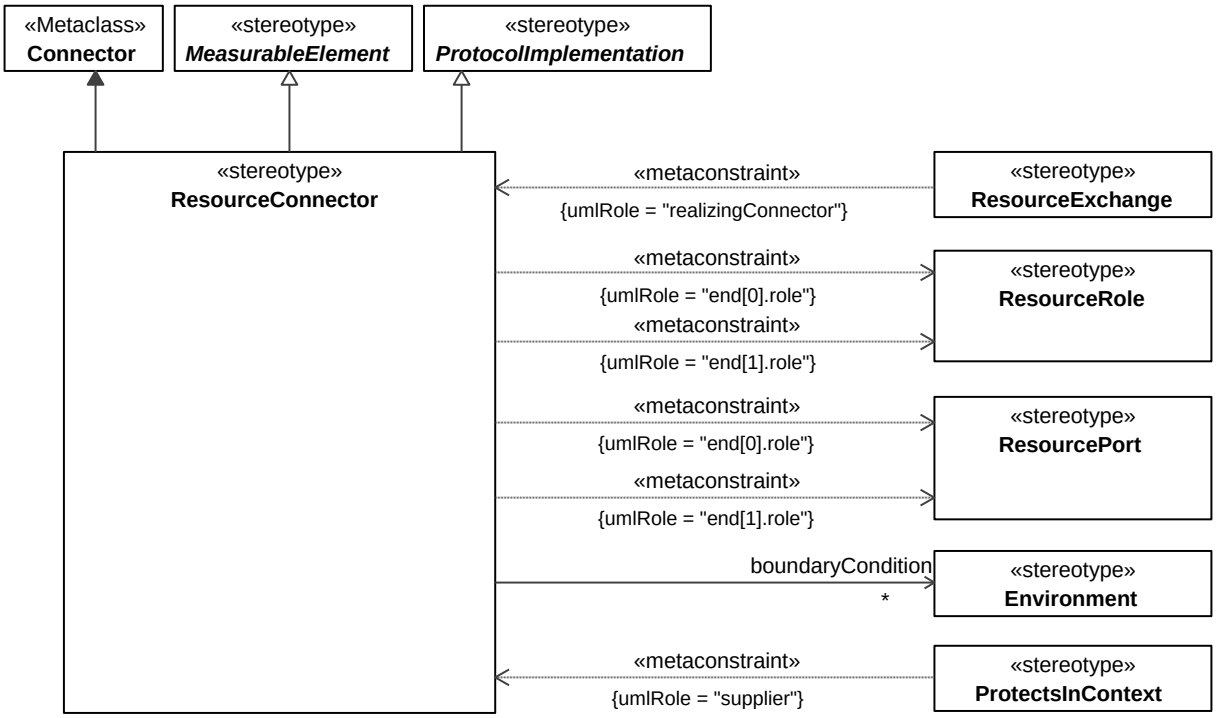

Figure 124 – ResourceConnector

Associations

boundaryCondition : Environment[*] Relates a ResourceConector to the extremes of the Environment in which it is required to be made available.

Constraints

[1] ResourceConnector.end The value for the role metaproperty for the owned ConnectorEnd must be stereotype «ResourcePort», «ResourceRole» or their specializations.

ResourceExchange

Package: Connectivity

isAbstract: No

Generalization: Exchange

Extension: InformationFlow

Description

Asserts that a flow can exist between ResourcePerformers (i.e. flows of data, people, materiel, or energy).

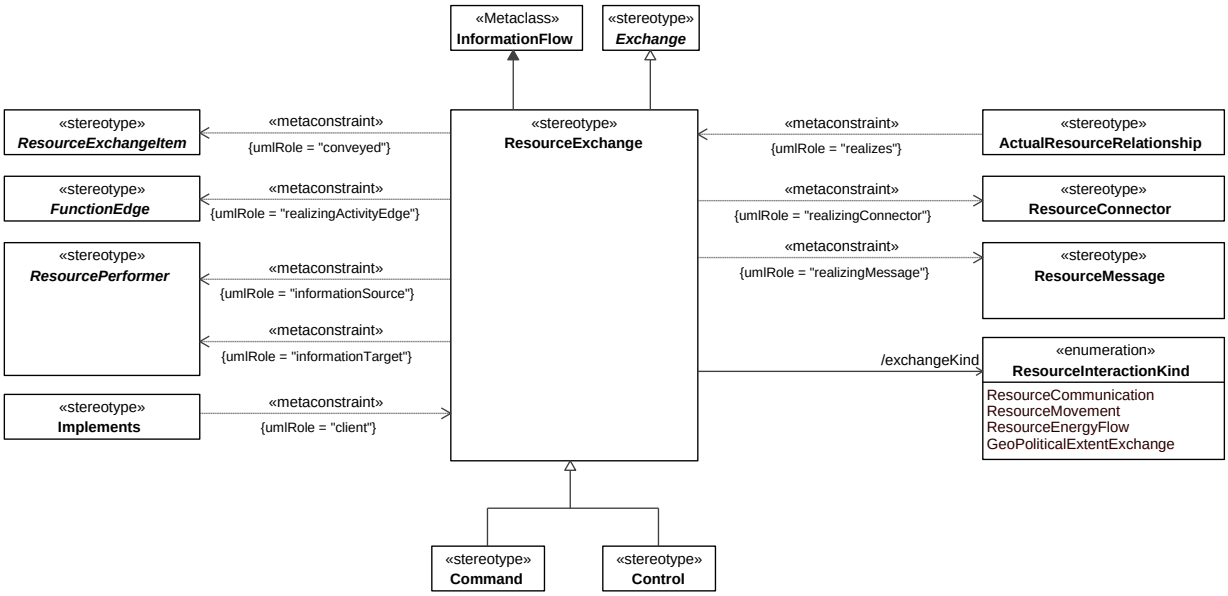

Figure 125 – ResourceExchange

Associations

exchangeKind : ResourceInteractionKind[] Captures the kind of ResourceExchange.

Constraints

[1] ResourceExchange.conveyed	In case of ResourceExchange.exchangeKind: = ResourceCommunication, the conveyed element must be stereotyped «DataElement» or its specializations, = ResourceMovement, the conveyed element must be stereotyped by the specialization of «PhysicalResource», = ResourceEnergyFlow, the conveyed element must be stereotyped «NaturalResource» or its specializations, = GeoPoliticalExtentExchange, the conveyed element must be stereotyped «GeoPoliticalExtentType» or its specializations.
[2] ResourceInteraction.informationSource	Value for the informationSource metaproperty must be stereotyped by the specialization of «ResourcePerformer».
[3] ResourceInteraction.informationTarget	Value for the informationTarget metaproperty must be stereotyped by the specialization of «ResourcePerformer».
[4] ResourceInteraction.realizingActivityEdge	Value for the realizingActivityEdge metaproperty must be stereotyped by the specialization of «FunctionEdge».
[5] ResourceInteraction.realizingConnector	Value for the realizingConnector metaproperty must be stereotyped «ResourceConnector» or its specializations.
[6] ResourceInteraction.realizingMessage	Value for the realizingMessage metaproperty must be stereotyped «ResourceMessage» or its specializations.

ResourceExchangeItem

Package: Connectivity

isAbstract: Yes

Generalization: Resource

Description

An abstract grouping for elements that defines the types of elements that can be exchanged between ResourcePerformers and conveyed by a ResourceExchange.

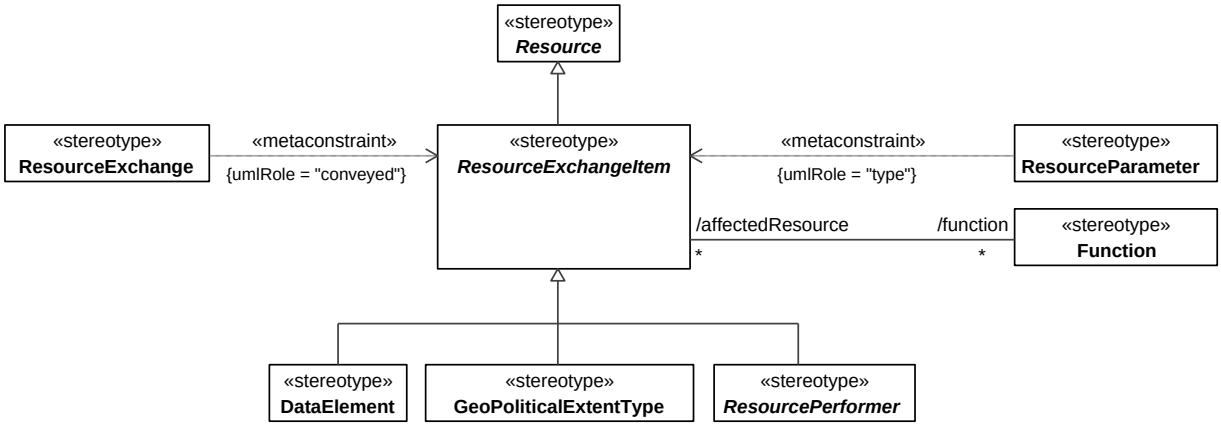

Figure 126 – ResourceExchangeItem

Associations

function : Function[*] Function using the ResourceExchangeItem internally.

ResourceInteractionKind

Package: Connectivity

isAbstract: No

Description

Enumeration of the possible kinds of resource exchange applicable to a ResourceExchange. Its enumeration literals are:

- ResourceCommunication - Indicates that the ResourceInteraction associated with the ResourceInteractionKind is a an implementation of logical flow of data between Resources.
- ResourceMovement - Indicates that the ResourceInteraction associated with the ResourceInteractionKind is a an implementation of logical flow of Resources between Resources.
- ResourceEnergyFlow - Indicates that the ResourceInteraction associated with the ResourceInteractionKind is a an implementation of logical flow of natural resources between Resources.
- GeoPoliticalExtentExchange - Indicates that the ResourceInteraction associated with the ResourceInteractionKind is a an implementation of logical flow where GeoPoliticalExtents (i.e. Borders) flow from one place to another.

ResourceInterface

Package: Connectivity

isAbstract: No

Generalization: PropertySet, InterfaceBlock

Extension: Class

Description

A declaration that specifies a contract between the ResourcePerformers it is related to and any other ResourcePerformers it can interact with. It is also intended to be an implementation of a specification of an Interface in the Business and/or Service layer.

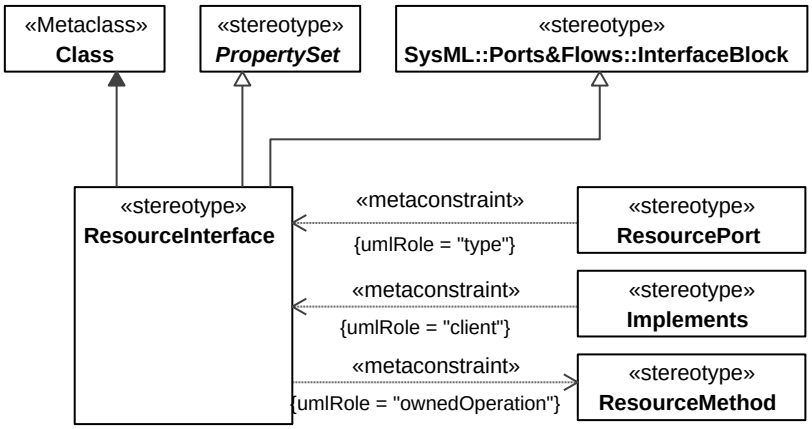

Figure 127 – ResourceInterface

Constraints

[1] ResourceInterface.ownedOperation — Values for ownedOperation metaproperty must be stereotyped «ResourceMethod» or its specializations.

7.1.8.4 UAF::Resources::Processes

Contains the elements that contribute to the Resources Processes Viewpoint.

Function

Package: Processes

isAbstract: No

Generalization: Activity, SubjectOfResourceConstraint

Extension: Activity

Description

An Activity which is specified in context of the ResourcePerformer (human or machine) that IsCapableOf Performing it.

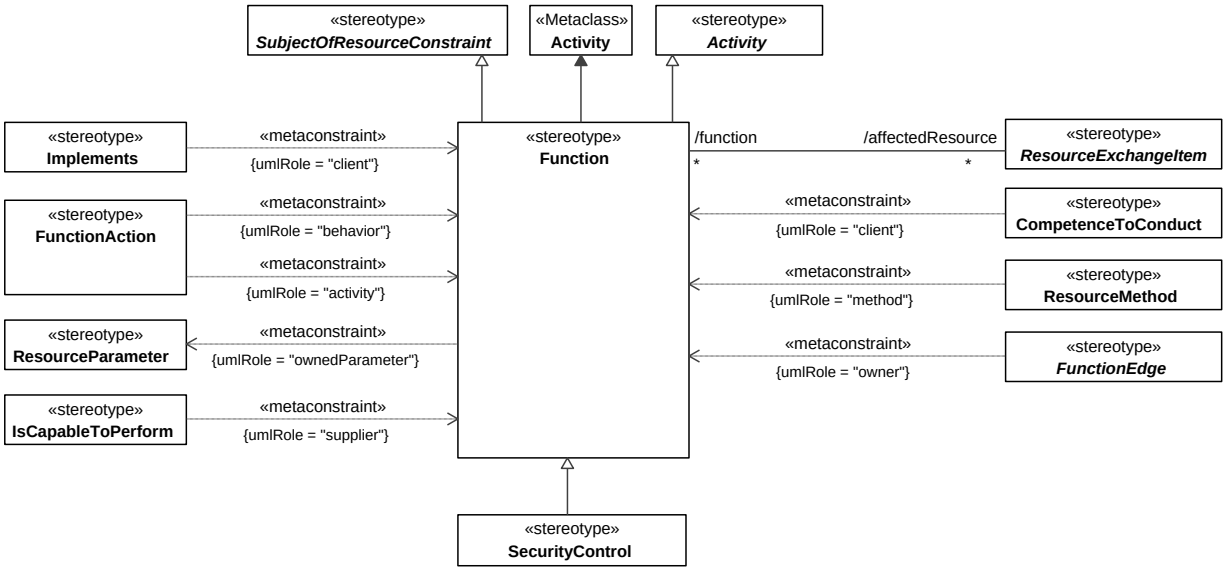

Figure 128 – Function

Associations

affectedResource : ResourceExchangeItem[*] — ResourceExchangeItems consumed and produced internally within a Function.

Constraints

[1] Function.ownedParameter	The values for the ownedParameter metaproperty must be stereotyped «ResourceParameter» or its specializations.

FunctionAction

Package: Processes

isAbstract: No

Generalization: MeasurableElement

Extension: CallBehaviorAction

Description

A call of a Function indicating that the Function is performed by a ResourceRole in a specific context.

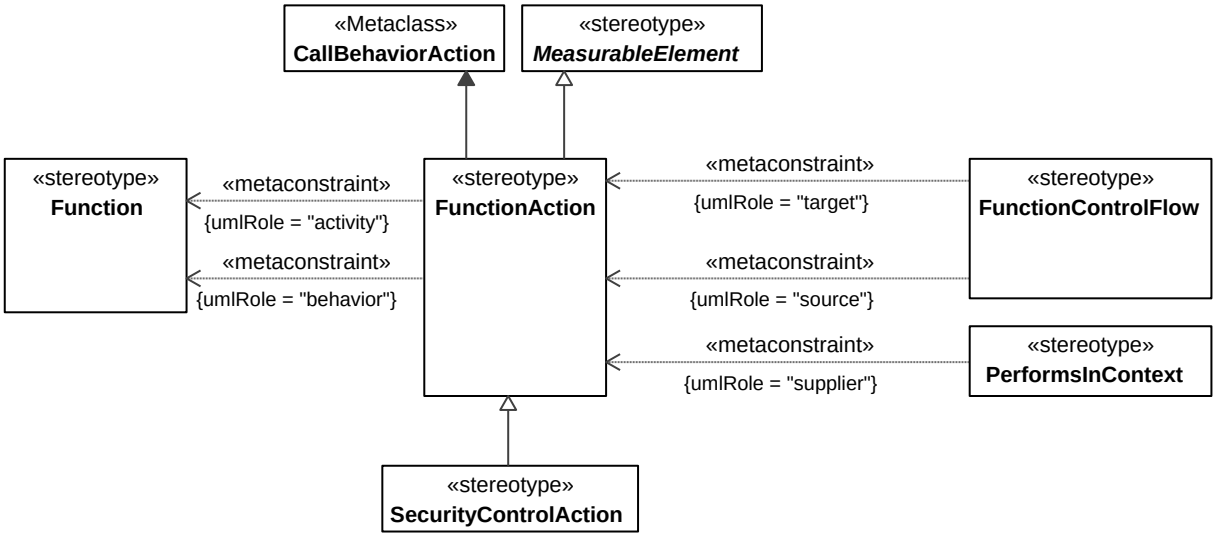

Figure 129 – FunctionAction

Constraints

[1] FunctionAction.activity	Value for the activity metaproperty must be stereotyped «Function» or its specializations.
[2] FunctionAction.behavior	Value for the behavior metaproperty must be stereotyped «Function» or its specializations.

FunctionControlFlow

Package: Processes

isAbstract: No

Generalization: FunctionEdge

Extension: ControlFlow

Description

An ActivityEdge that shows the flow of control between FunctionActions.

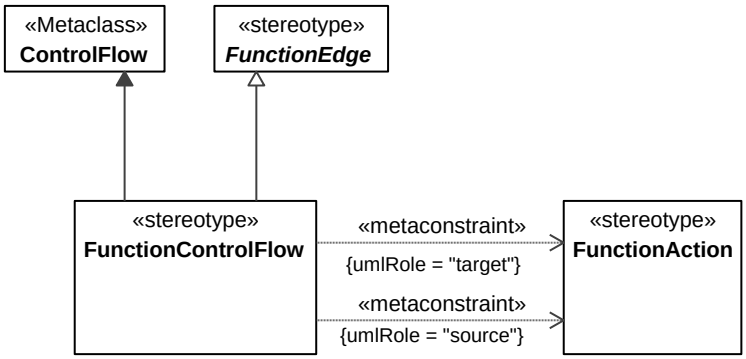

Figure 130 – FunctionControlFlow

Constraints

[1] FunctionControlFlow.source	Value for the source metaproperty must be stereotyped «FunctionAction» or its specializations.
[2] FunctionControlFlow.target	Value for the target metaproperty must be stereotyped «FunctionAction» or its specializations.

FunctionEdge

Package: Processes

isAbstract: Yes

Generalization: MeasurableElement

Extension: ActivityEdge

Description

Abstract grouping for FunctionControlFlow and FunctionObjectFlow.

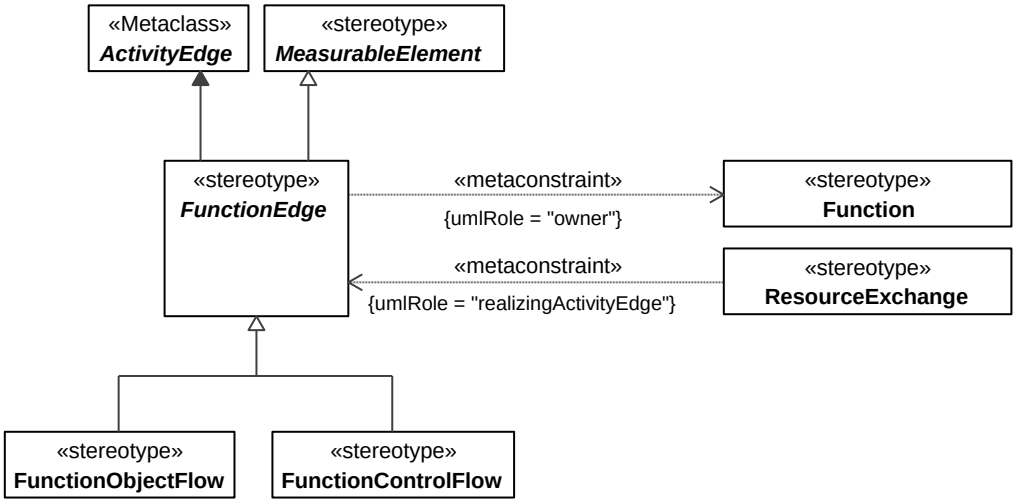

Figure 131 – FunctionEdge

Constraints

[1] FunctionEdge.owner	«FunctionEdge» must be owned directly or indirectly by «Function» or its specializations.

FunctionObjectFlow

Package: Processes

isAbstract: No

Generalization: FunctionEdge

Extension: ObjectFlow
Description

An ActivityEdge that shows the flow of Resources (objects/data) between FunctionActions.

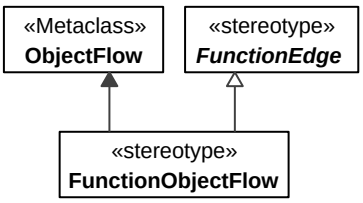

Figure 132 – FunctionObjectFlow

7.1.8.5 UAF::Resources::States

Contains the elements that contribute to the Resources States Viewpoint.

ResourceStateDescription

Package: States

isAbstract: No

Generalization: MeasurableElement

Extension: StateMachine
Description

A state machine describing the behavior of a ResourcePerformer, depicting how the ResourcePerformer responds to various events and the actions.

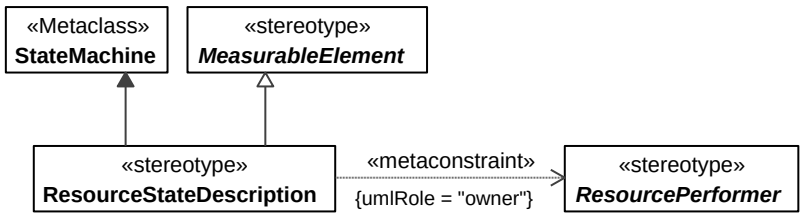

Figure 133 – ResourceStateDescription

Constraints

[1] ResourceStateDescription.owner	Values for the owner metaproperty must be stereotyped with the specialization of «ResourcePerformer».

7.1.8.6 UAF::Resources::Interaction Scenarios

Contains the elements that contribute to the Resources Interaction Scenarios Viewpoint.

ResourceMessage

Package: Interaction Scenarios

isAbstract: No

Generalization: MeasurableElement

Extension: Message
Description

Message for use in an Resource Event-Trace which carries any of the subtypes of ResourceExchange.

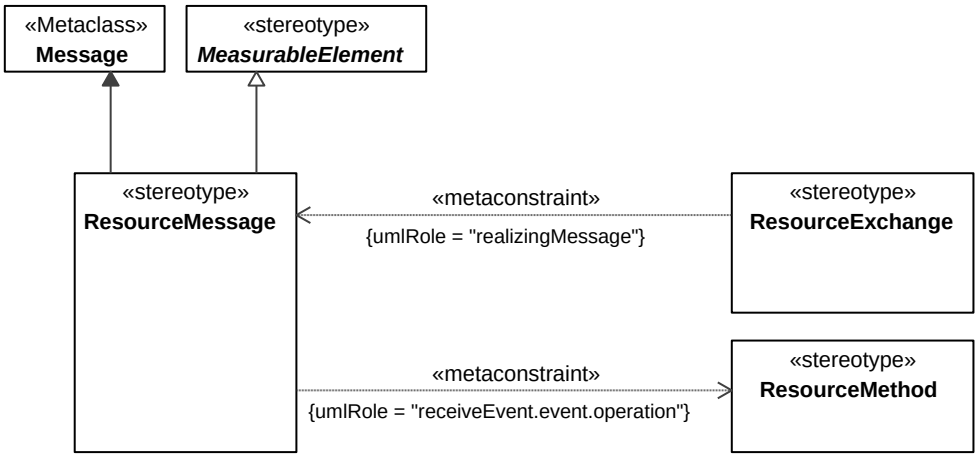

Figure 134 – ResourceMessage

Constraints

[1] ResourceMessage.receiveEvent.event.operation — Values for the receiveEvent.event.operation metaproperty must be stereotyped with «ResourceMethod» or its specializations.

7.1.8.7 UAF::Resources::Information

Contains the elements that contribute to the Resources Information Viewpoint.

DataElement

Package: Information

isAbstract: No

Generalization: ResourceExchangeItem, SubjectOfResourceConstraint, Asset

Extension: Class

Description

A formalized representation of data that is managed by or exchanged between systems.

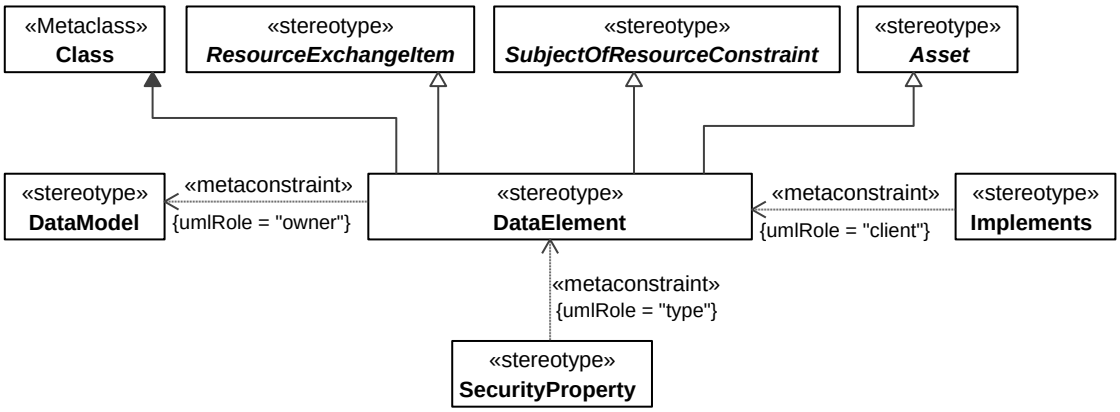

Figure 135 – DataElement

Constraints

[1] DataElement.owner — Values for the owner metaproperty must be stereotyped «DataModel» or its specializations.

DataModel

Package: Information

isAbstract: No

Generalization: SubjectOfOperationalConstraint

Extension: Package
Description

A structural specification of data types, showing relationships between them that is devoid of implementation detail. The type of data captured in the DataModel is described using the enumaration DataModelKind (Conceptual,Logical and Physical).

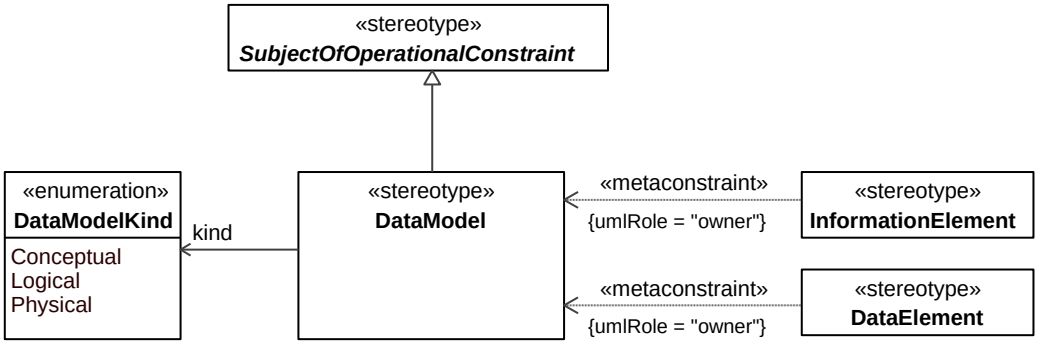

Figure 136 – DataModel

Associations

kind : DataModelKind[] Captures the kind of DataModel being respresented, Conceptual, Logical or Physcial.

DataModelKind

Package: Information

isAbstract: No
Description

Enumeration of the possible kinds of DataModel. Its enumeration literals are:

- Conceptual - Indicates that the DataModel associated with the DataModelKind is a conceptual DataModel that defines the required high-level data concepts and their relationships.
- Logical - Indicates that the DataModel associated with the DataModelKind is a logical data model that allows analysis of an architecture's data definition aspect, without consideration of implementation specific or product specific issues. It details the conceptual data model.
- Physical - Indicates that the DataModel associated with the DataModelKind is a physical data model that is an implementable specification of a data structure. A physical data model realizes a logical data model, taking into account implementation restrictions and performance issues while still enforcing the constraints, relationships and typing of the logical data model.

7.1.8.8 UAF::Resources::Constraints

Contains the elements that contribute to the Resources Constraints Viewpoint.

ResourceConstraint

Package: Constraints

isAbstract: No

Generalization: Rule

Extension: Constraint
Description

A rule governing the structural or functional aspects of an implementation.

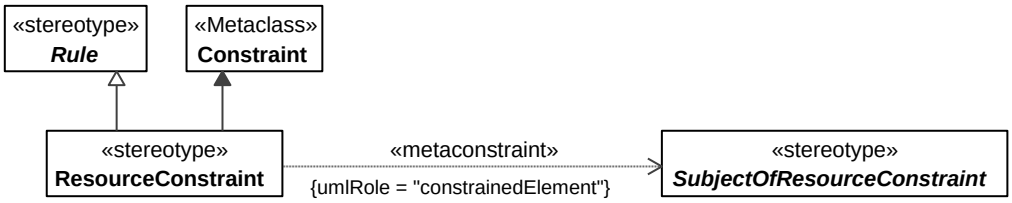

Figure 137 – ResourceConstraint

Constraints

[1] ResourceConstraint.constrainedElement	Value for the constrainedElement metaproperty must be stereotyped by the specialization of «SubjectOfResourceConstraint».

SubjectOfResourceConstraint

Package: Constraints

isAbstract: Yes

Generalization: UAFElement

Extension: Element

Description

An abstract grouping of elements that can be the subject of a ResourceConstraint.

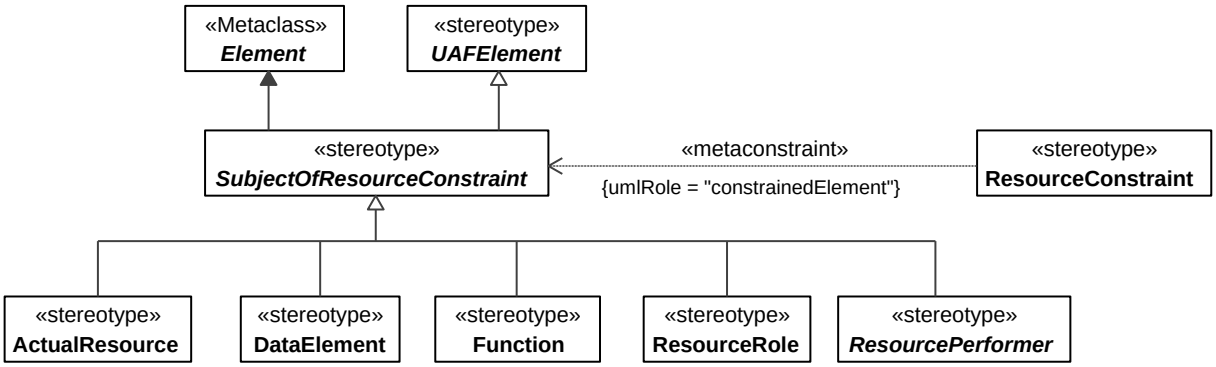

Figure 138 – SubjectOfResourceConstraint

7.1.8.9 UAF::Resources::Roadmap

Contains the elements that contribute to the Resources Roadmap Viewpoint.

Forecast

Package: Roadmap

isAbstract: No

Generalization: MeasurableElement

Extension: Dependency

Description

A dependency relationship that specifies a transition from one Asset, Standard, Competence to another future one. It is related to an ActualEnterprisePhase to give it a temporal context.

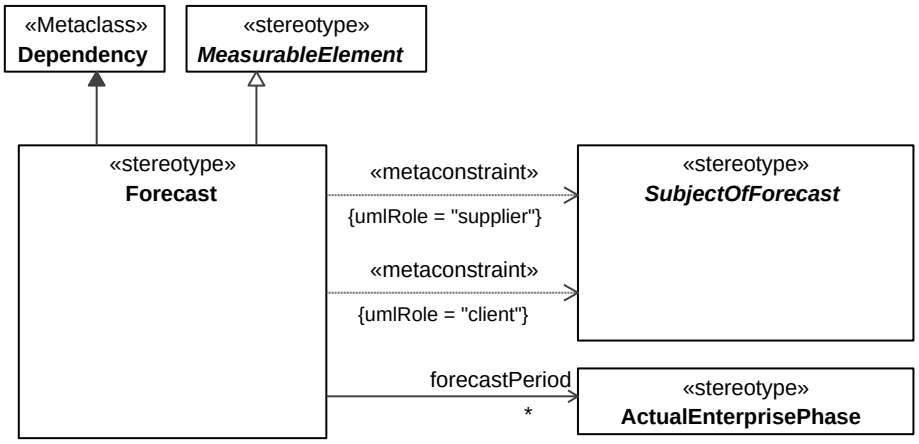

Figure 139 – Forecast

Associations

forecastPeriod : ActualEnterprisePhase[*] Relates the SubjectOfForecast to the ActualEnterprisePhase in which the SubjectOfForecast is expected to be provided.

Constraints

[1] Forecast.client Value for the client metaproperty must be stereotyped by the specialization of «SubjectOfForecast».

[2] Forecast.pair Values for the client and supplier metaproperties must be stereotyped by the same specialization of «SubjectOfForecast» (e.g. «Software» to «Software», «Standard» to «Standard», etc).

[3] Forecast.supplier Value for the supplier property must be stereotyped by the specialization of «SubjectOfForecast».

SubjectOfForecast

Package: Roadmap

isAbstract: Yes

Generalization: UAFElement

Extension: Class

Description

An abstract grouping of elements that can be the subject of a Forecast.

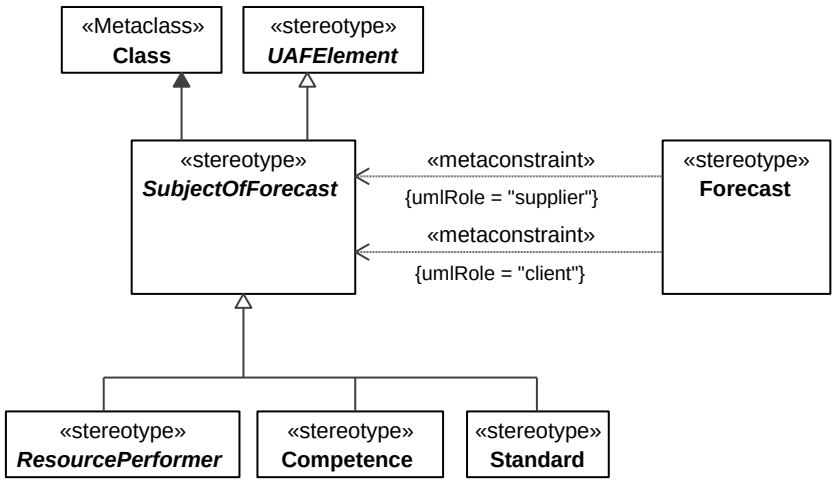

Figure 140 – SubjectOfForecast

Technology

Package: Roadmap

isAbstract: No

Generalization: ResourceArtifact

Extension: Class
Description

A sub type of ResourceArtifact that indicates a technology domain, i.e. nuclear, mechanical, electronic, mobile telephony etc.

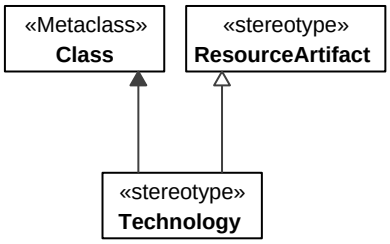

Figure 141 – Technology

VersionedElement

Package: Roadmap

isAbstract: Yes

Generalization: UAFElement

Extension: Class
Description

An abstract grouping of ResourcePerformer and ServiceSpecification that allows VersionOfConfiguration to be related to ActualProjectMilestones.

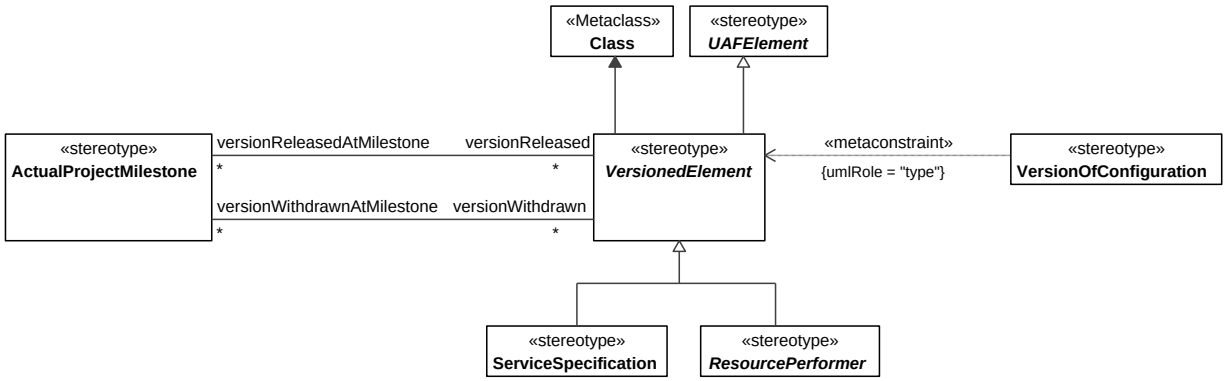

Figure 142 – VersionedElement

Associations

versionReleasedAtMilestone : ActualProjectMilestone[*]	Relates a VersionedElement to the ActualProjectMilestone. It indicates the ActualProjectMilestone at which the VersionedElement is released.
versionWithdrawnAtMilestone : ActualProjectMilestone[*]	Relates a VersionedElement to the ActualProjectMilestone. It indicates the ActualProjectMilestone at which the VersionedElement is withdrawn.

VersionOfConfiguration

Package: Roadmap

isAbstract: No

Generalization: MeasurableElement

Extension: Property
Description

A property of a WholeLifeConfiguration, used in version control of a VersionedElement. It asserts that a VersionedElement is a version of a WholeLifeConfiguration.

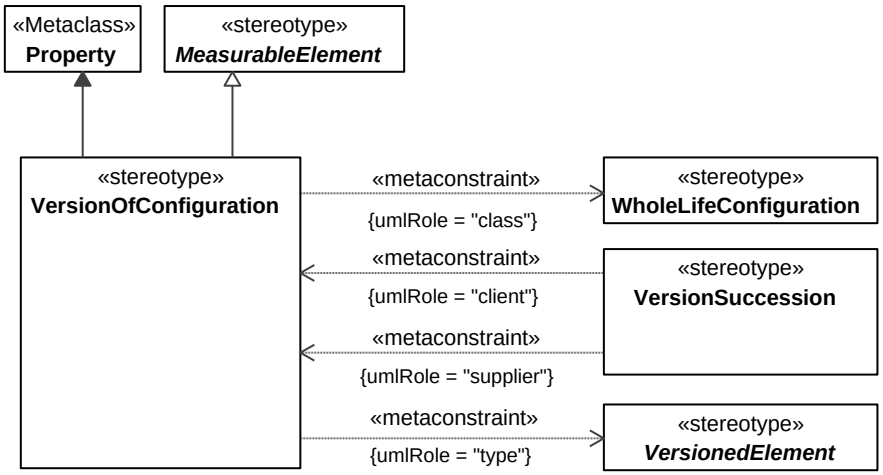

Figure 143 – VersionOfConfiguration

Constraints

[1] VersionOfConfiguration.class	Value for the class metaproperty must be stereotyped «WholeLifeConfiguration» or its specializations.
[2] VersionOfConfiguration.type	Value for the type metaproperty must be stereotyped by the specialization of «VersionedElement».

VersionSuccession

Package: Roadmap

isAbstract: No

Generalization: MeasurableElement

Extension: Dependency
Description

A dependency relationship between two VersionOfConfigurations that denotes that one VersionOfConfiguration follows from another.

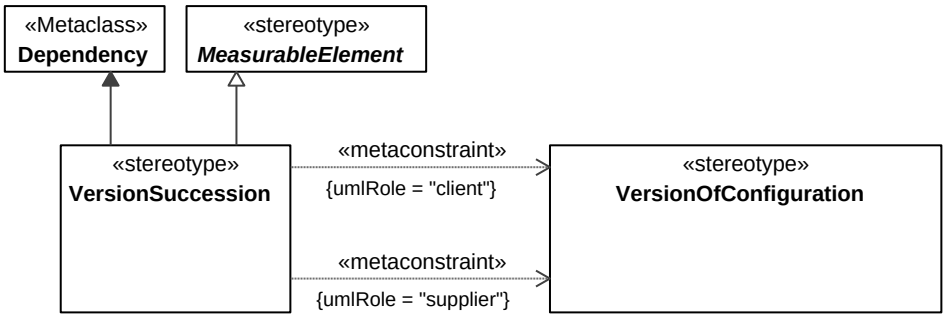

Figure 144 – VersionSuccession

Constraints

[1] VersionSuccession.client	Value for the client metaproperty must be stereotyped «VersionOfConfiguration» or its specializations.
[2] VersionSuccession.supplier	Value for the supplier metaproperty must be stereotyped «VersionOfConfiguration» or its specializations.

WholeLifeConfiguration

Package: Roadmap

isAbstract: No

Generalization: PropertySet, Block

Extension: Class

Description

A set of VersionedElements.

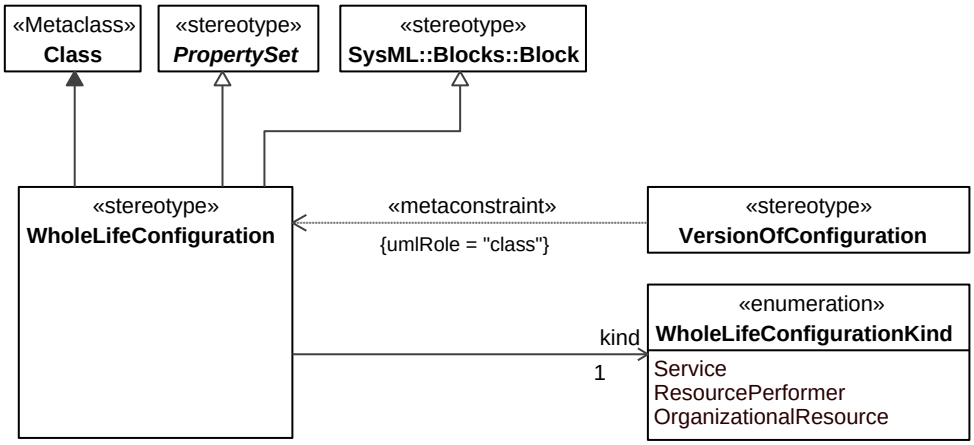

Figure 145 – WholeLifeConfiguration

Associations

kind : WholeLifeConfigurationKind[1] Captures the kind of WholeLifeConfiguration.

WholeLifeConfigurationKind

Package: Roadmap

isAbstract: No

Description

Enumeration of the possible kinds of WholeLifeConfiguration. Its enumeration literals are:

- Service - Indicates that the WholeLifeConfiguration associated with the WholeLifeConfigurationKind is the master specification from which Services are versioned.
- ResourcePerformer - Indicates that the WholeLifeConfiguration associated with the WholeLifeConfigurationKind is the master specification from which ResourcePerformers are versioned.
- OrganizationalResource - Indicates that the WholeLifeConfiguration associated with the WholeLifeConfigurationKind is the master specification from which OrganizationalResources are versioned.

7.1.8.10 UAF::Resources::Traceability

Contains the elements that contribute to the Resources Traceability Viewpoint.

ProtocolImplementation

Package: Traceability

isAbstract: Yes

Generalization: UAFElement

Extension: Element
Description

An abstract grouping of architectural elements that can implement Protocols.

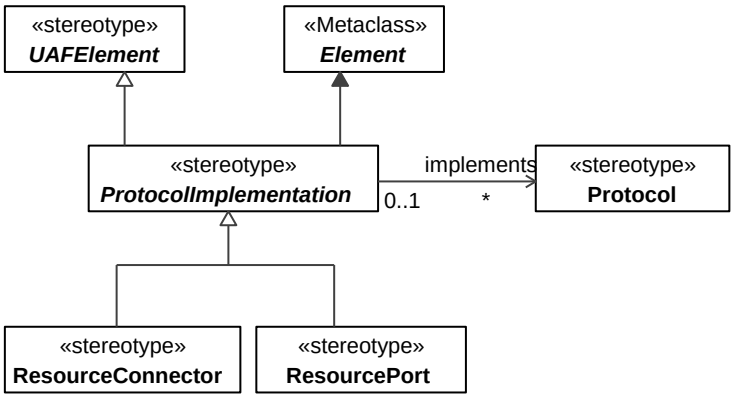

Figure 146 – ProtocolImplementation

Associations

implements : Protocol[*] Relates the ResourceConnector and ResourcePort to the Protocols that they can implement.

7.1.9 UAF::Security

Stakeholders: Security Architects, Security Engineers. Systems Engineers, Operational Architects.
Concerns: addresses the security constraints and information assurance attributes that exist on exchanges between resources and OperationalPerformers
Definition: illustrates the security assets, security constraints, security controls, families, and measures required to address specific security concerns.

7.1.9.1 UAF::Security::Taxonomy

Contains the elements that contribute to the Security Taxonomy Viewpoint.

Asset

Package: Taxonomy

isAbstract: Yes

Generalization: ConceptItem, PropertySet, LocationHolder, SubjectOfSecurityConstraint, Block

Extension: Class
Description

Asset as applied to Security views, an abstract element that indicates the types of elements that can be considered as a subject for security analysis.

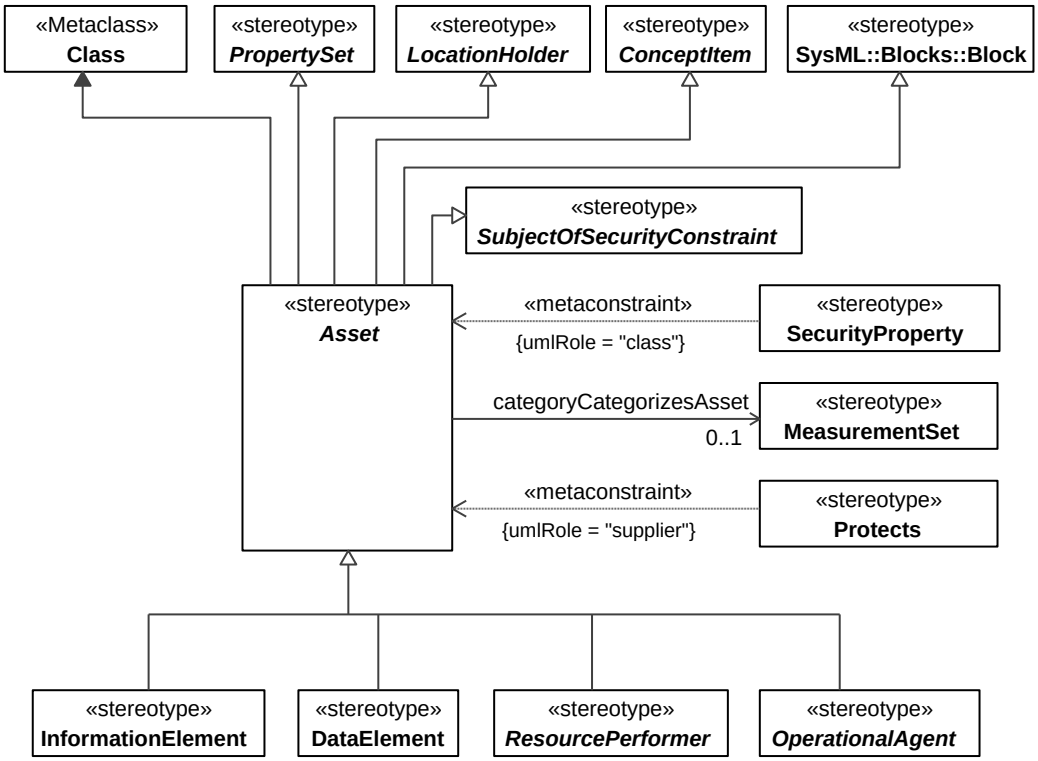

Figure 147 – Asset

Associations

categoryCategorizesAsset : MeasurementSet[0..1] Enables association of an Asset to the set of security related measurements (MeasurementSet).

OperationalMitigation

Package: Taxonomy

isAbstract: No

Generalization: OperationalArchitecture

Extension: Class

Description

A set of security measures intended to address against specific cyber risks. Comprises a subset of SecurityControls that are required to protect the asset at OperationalPerformer (OperationalRole).

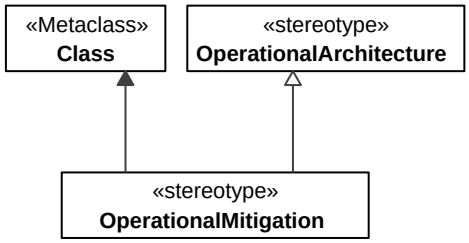

Figure 148 – OperationalMitigation

ResourceMitigation

Package: Taxonomy

isAbstract: No

Generalization: ResourceArchitecture

Extension: Class
Description

A set of security measures intended to address specific cyber risks. Comprises a subset of TailoredSecurityControls that are used to protect the asset at resource (ResourceRole).

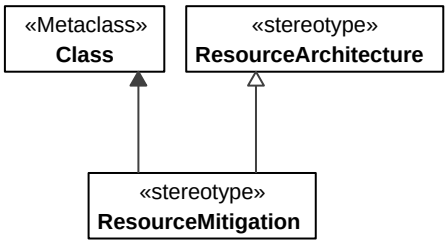

Figure 149 – ResourceMitigation

SecurityEnclave

Package: Taxonomy

isAbstract: No

Generalization: ResourceArchitecture

Extension: Class
Description

Collection of information systems connected by one or more internal networks under the control of a single authority and security policy. The systems may be structured by physical proximity or by function, independent of location.

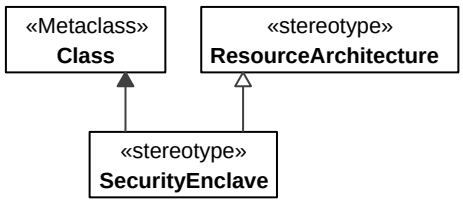

Figure 150 – SecurityEnclave

7.1.9.2 UAF::Security::Structure

Contains the elements that contribute to the Security Structure Viewpoint.

AssetRole

Package: Structure

isAbstract: Yes

Generalization: UAFElement

Extension: Element
Description

AssetRole as applied to Security views, an abstract element that indicates the type of elements that can be considered as a subject for security analysis in the particular context.

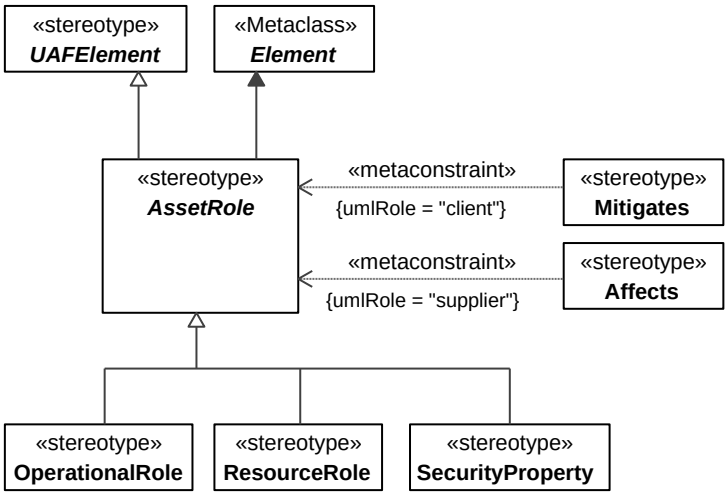

Figure 151 – AssetRole

SecurityProperty

Package: Structure

isAbstract: No

Generalization: MeasurableElement, AssetRole

Extension: Property

Description

SecurityProperty is used to assign an aggregated security marking (from the SecurityAttributes enumerated list: ClassificationType) to designate this "aggregated" security classification. The inter-connectivity of different data sets may allow more sensitive connections to be made by association. Aggregation, accumulation and association of data (within ICT systems and on removable media) must be carefully considered as part of the risk management process as additional protective controls may or may not be appropriate. Aggregation does not automatically trigger an increase in protective marking. For instance, a database with thousands of records which are individually OFFICIAL should not be relabeled as a SECRET database. Instead, information owners are expected to make decisions about controls based on a risk assessment, and should consider what the aggregated information is, who needs to access it, and how.

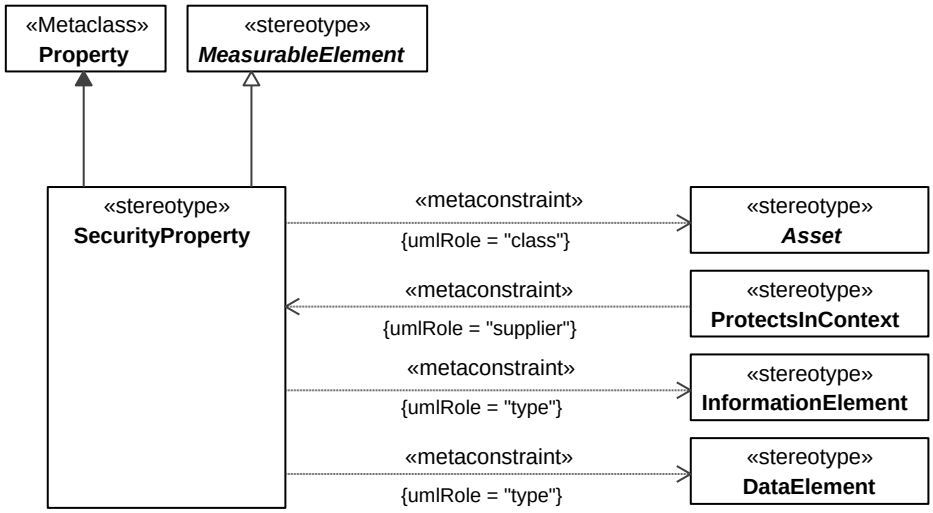

Figure 152 – SecurityProperty

Constraints

[1] SecurityProperty.class Value for the class metaproperty must be stereotyped by the specialization of «Asset».

[2] SecurityProperty.type	In case of value for the class metaproperty is stereotyped: a. by any of specializations of «OperationalAgent», values for the type metaproperty must be stereotyped «InformationElement» or its specializations, b. by any of specializations of «ResourcePerformer», values for the type metaproperty must be stereotyped «DataElement» or its specializations, c. «InformationElement», values for the type metaproperty must be stereotyped «InformationElement» or its specializations, d. «DataElement», values for the type metaproperty must be stereotyped «DataElement» or its specializations.

7.1.9.3 UAF::Security::Processes

Contains the elements that contribute to the Security Processes Viewpoint.

EnhancedSecurityControl

Package: Processes

isAbstract: No

Generalization: SecurityControl

Extension: Activity
Description

A type of Activity that specifies a safeguard or countermeasure prescribed for a ResourcePerformer. It is intended to protect the confidentiality, integrity, and availability of the Resource's information and to meet a set of defined security requirements.

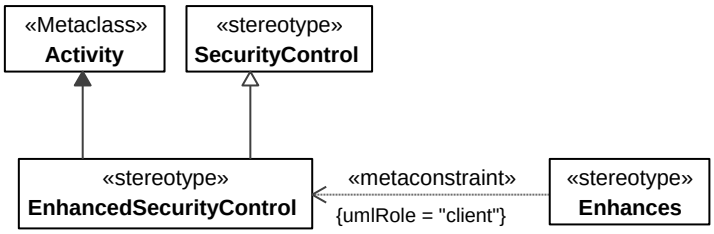

Figure 153 – EnhancedSecurityControl

Enhances

Package: Processes

isAbstract: No

Generalization: MeasurableElement

Extension: Dependency
Description

A dependency relationship relating the EnhancedSecurityControl to a SecurityControl.

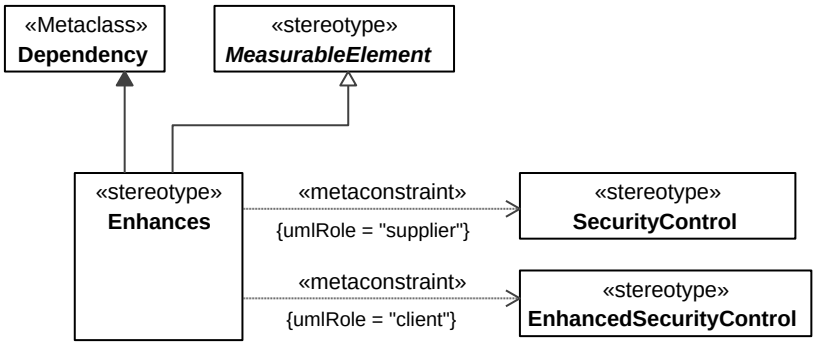

Figure 154 – Enhances

Constraints

[1] Enhances.client Value for the client metaproperty must be stereotyped «EnhancedSecurityControl» or its specializations.

[2] Enhances.supplier Value for the supplier metaproperty must be stereotyped «SecurityControl» or its specializations.

Protects

Package: Processes

isAbstract: No

Generalization: MeasurableElement

Extension: Dependency

Description

A dependency that asserts that a SecurityControl is required to protect an Asset.

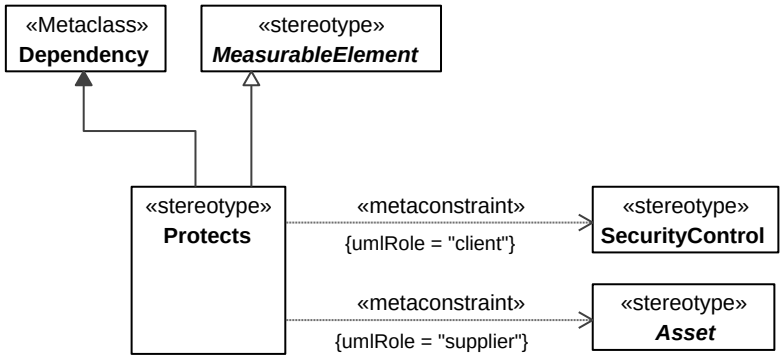

Figure 155 – Protects

Constraints

[1] Protects.client Value for the client metaproperty must be stereotyped «SecurityControl» or its specializations.

[2] Protects.supplier Value for the supplier metaproperty must be stereotyped bu the specialization of «Asset».

ProtectsInContext

Package: Processes

isAbstract: No

Generalization: MeasurableElement

Extension: Dependency

Description

A dependency relationship that relates a SecurityControlAction to a OperationalRole, or a ResourceRole. It indicates that SecurityControl is required to protect an Asset in a specific context or configuration.

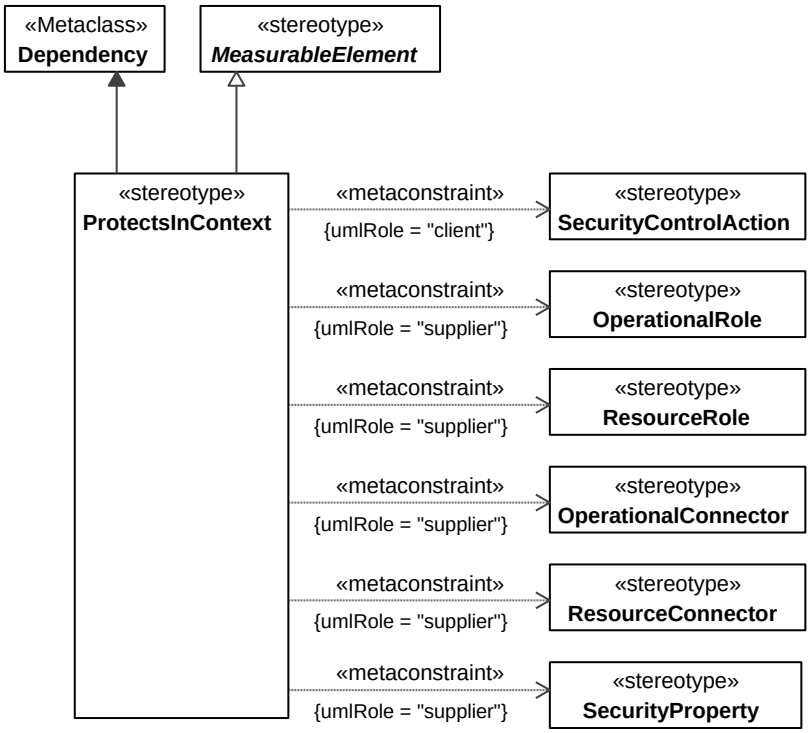

Figure 156 – ProtectsInContext

Constraints

[1] ProtectsInContext.client	Value for the client metaproperty must be stereotyped «SecurityControlAction» or its specializations.
[2] ProtectsInContext.supplier	Value for the supplier metaproperty must be stereotyped «OperationalRole», «ResourceRole», «OperationalConnector», «ResourceConnector», «SecurityProperty» or their specializations.

SecurityControl

Package: Processes

isAbstract: No

Generalization: OperationalActivity, Function

Extension: Activity

Description

A type of Activity. A safeguard or countermeasure prescribed for an information system or an organization designed to protect the confidentiality, integrity, and availability of the asset's information and to meet a set of defined security requirements.

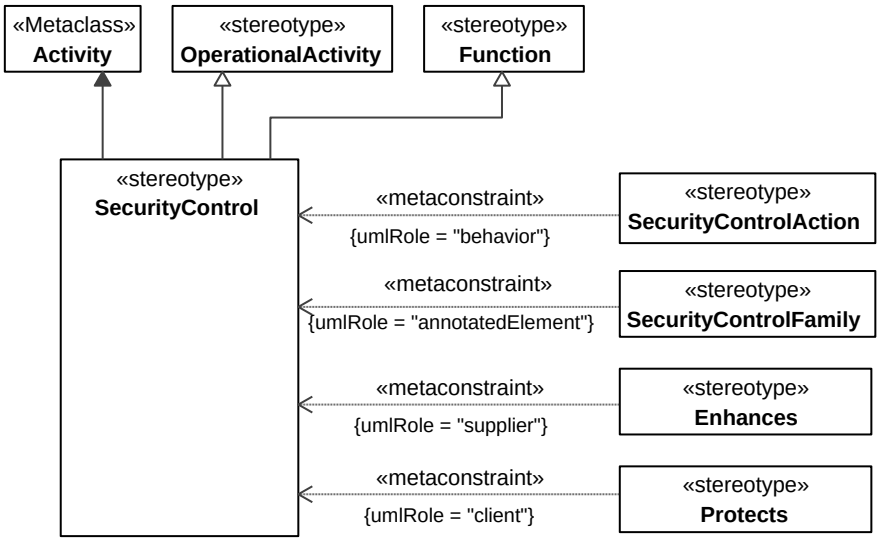

Figure 157 – SecurityControl

SecurityControlAction

Package: Processes

isAbstract: No

Generalization: OperationalActivityAction, FunctionAction

Extension: CallBehaviorAction

Description

A call of a SecurityControl in the context of another SecurityControl. It is used to show how a set of SecurityControls can be used to protect an asset in a particular context.

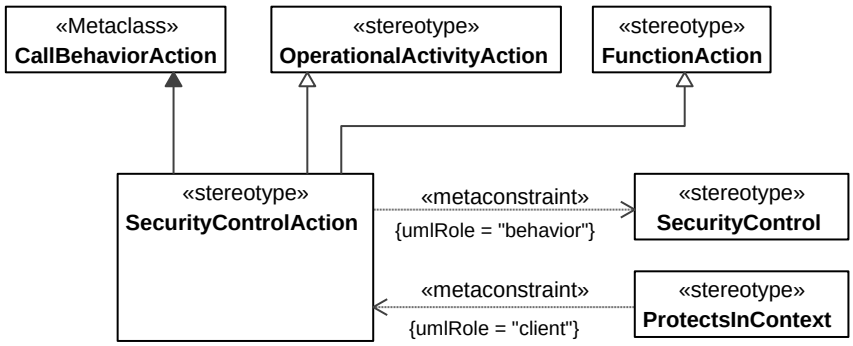

Figure 158 – SecurityControlAction

Constraints

[1] SecurityControlAction.behavior	Value for behavior metaproperty must be stereotyped «SecurityControl» or its specializations.

SecurityControlFamily

Package: Processes

isAbstract: No

Generalization: ElementGroup, MeasurableElement

Extension: Comment

Description

An element that organizes security controls into a family.

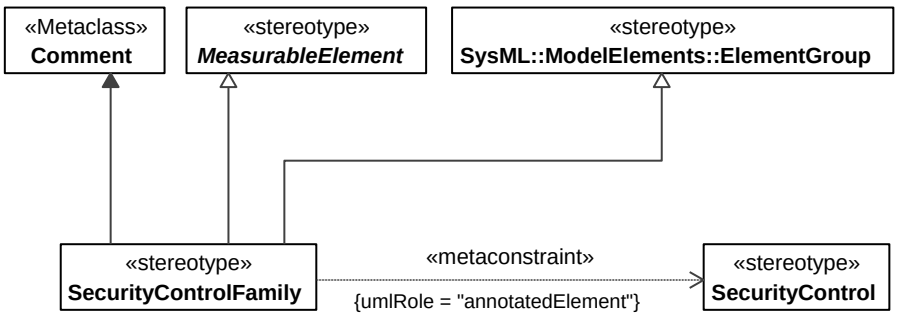

Figure 159 – SecurityControlFamily

Constraints

[1] SecurityControlFamily.annotatedElement	Value for the annotatedElement metaproperty must be stereotyped «SecurityControl» or its specializations.

7.1.9.4 UAF::Security::Constraints

Contains the elements that contribute to the Security Constraints Viewpoint.

ActualRisk

Package: Constraints

isAbstract: No

Generalization: ActualPropertySet

Extension: InstanceSpecification

Description

An instance of a Risk. A value holder for Risk Measurements.

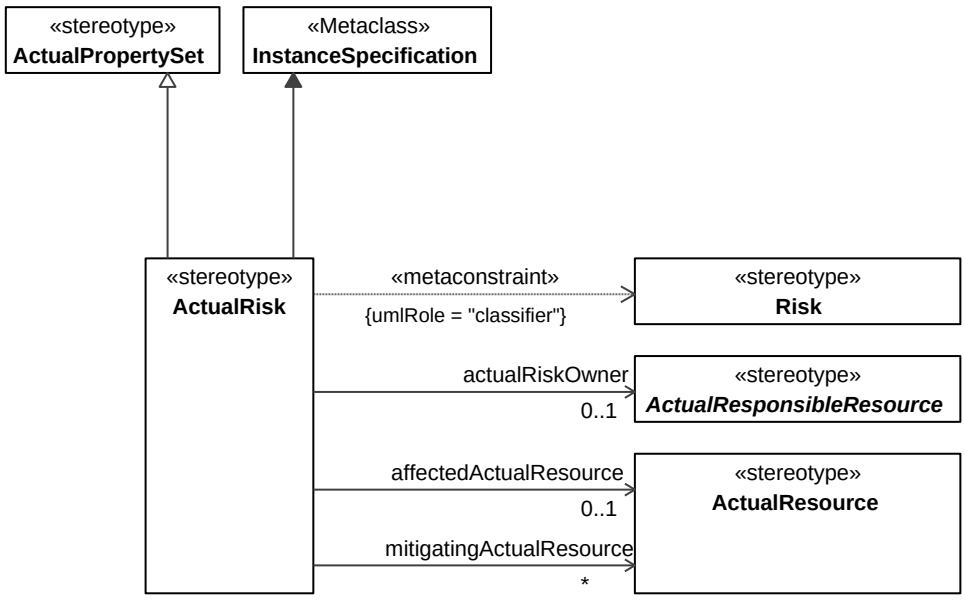

Figure 160 – ActualRisk

Associations

actualRiskOwner : ActualResponsibleResource[0..1]	Enables association of an ActualRisk to an actual organizational role that is responsible for executing the actual mitigation.
affectedActualResource : ActualResource[0..1]	Asserts that an ActualRisk is applicable to an ActualResource.

mitigatingActualResource : ActualResource[*]	Relates an actual mitigation (an ActualResource for mitigating a Risk) to an ActualRisk.

Risk

Package: Constraints

isAbstract: No

Generalization: PropertySet, Block

Extension: Class

Description

A statement of the impact of an event on Assets. It represents a constraint on an Asset in terms of adverse effects, with an associated measure. The measure is used to capture the extent to which an entity is threatened by a potential circumstance or event. Risk is typically a function of: (i) the adverse impacts that would arise if the circumstance or event occurs; and (ii) the likelihood of occurrence. Software related security risks are those risks that arise from the loss of confidentiality, integrity, or availability of information or information systems.

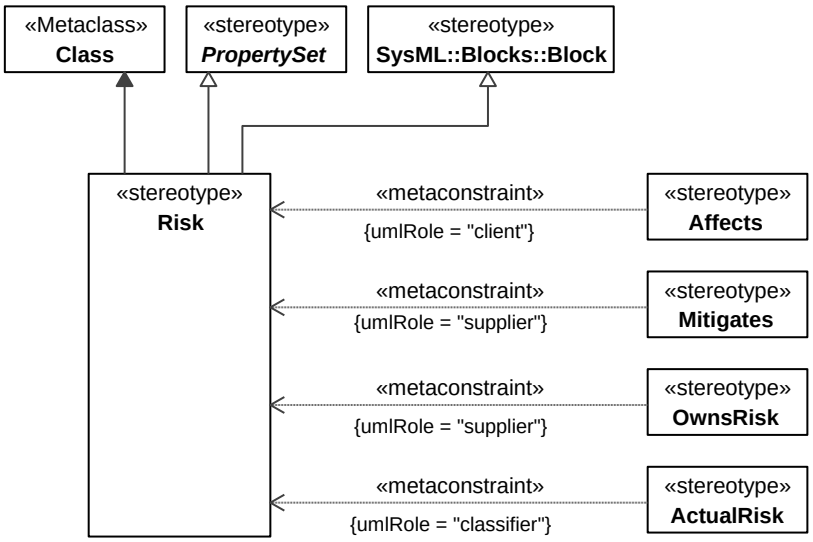

Figure 161 – Risk

SecurityConstraint

Package: Constraints

isAbstract: No

Generalization: Rule

Extension: Constraint

Description

A type of rule that captures a formal statement to define security laws, regulations, guidances, and policy.

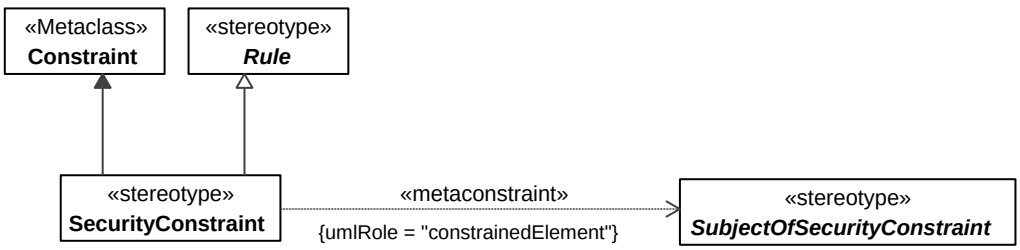

Figure 162 – SecurityConstraint

Constraints

[1] Security.constrainedElement Value for the constrainedElement metaproperty must be stereotyped by the specialization of «SubjectOfSecurityConstraint».

SubjectOfSecurityConstraint

Package: Constraints

isAbstract: Yes

Generalization: UAFElement

Extension: Element

Description

An abstract grouping of elements that can be the subject of a SecurityConstraint.

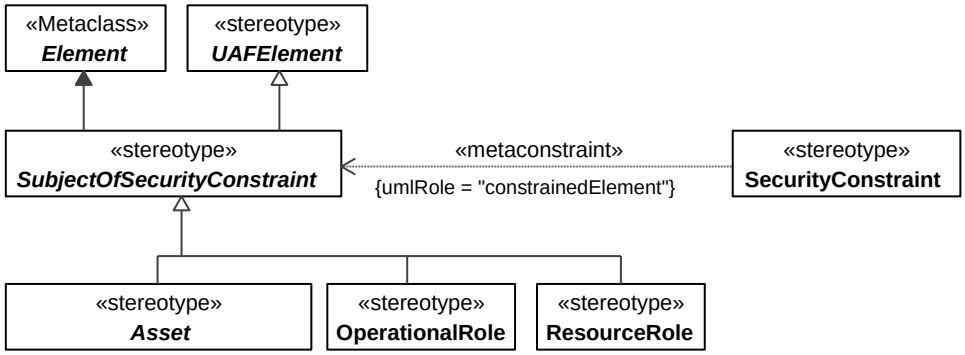

Figure 163 – SubjectOfSecurityConstraint

7.1.9.5 UAF::Security::Traceability

Contains the elements that contribute to the Security Traceability Viewpoint.

Affects

Package: Traceability

isAbstract: No

Generalization: MeasurableElement

Extension: Dependency

Description

A dependency that asserts that a Risk is applicable to an AssetRole.

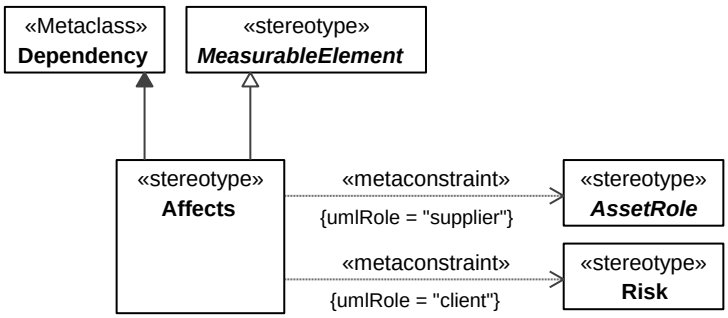

Figure 164 – Affects

Constraints

[1] Affects.client Value for the client metaproperty must be stereotyped «Risk» or its specializations.

[2] Affects.supplier Value for the supplier metaproperty must be stereotyped «AssetRole» or its specializations.

Mitigates

Package: Traceability

isAbstract: No

Generalization: MeasurableElement

Extension: Dependency
Description

A dependency relating an operational or resource mitigation to a Risk.

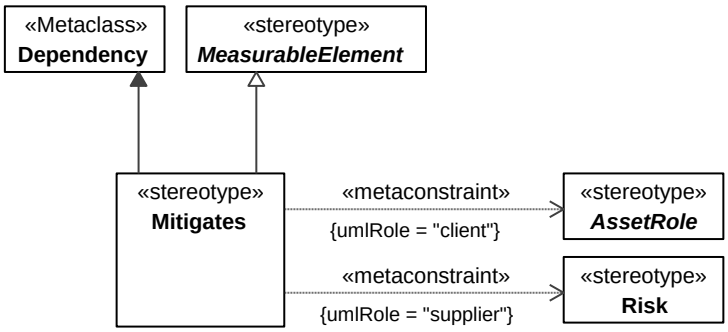

Figure 165 – Mitigates

Constraints

[1] Mitigates.client Value for the client metaproperty must be stereotyped «AssetRole» or its specializations.

[2] Mitigates.supplier Value for the supplier metaproperty must be stereotyped «Risk» or its specializations.

OwnsRisk

Package: Traceability

isAbstract: No

Generalization: MeasurableElement, Allocate

Extension: Abstraction
Description

An abstraction relating a Risk to an organizational role that is responsible for executing the risk mitigation package.

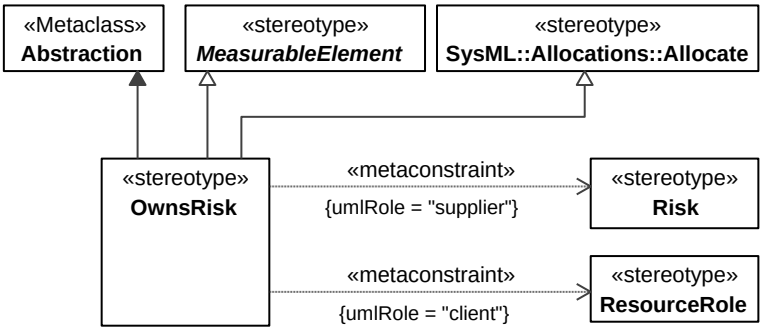

Figure 166 – OwnsRisk

Constraints

[1] OwnsRisk.client Value for the client metaproperty must be stereotyped «ResourceRole» or its specializations.

[2] OwnsRisk.supplier Value for the supplier metaproperty must be stereotyped «Risk» or its specializations.

7.1.10 UAF::Project

Stakeholders: PMs, Project Portfolio Managers, Enterprise Architects.
Concerns: project portfolio, projects and project milestones.

Definition: describes projects and project milestones, how those projects deliver capabilities, the organizations contributing to the projects and dependencies between projects.

7.1.10.1 UAF::Project::Taxonomy

Contains the elements that contribute to the Project Taxonomy Viewpoint.

ActualMilestoneKind

Package: Taxonomy

isAbstract: No
Description

Enumeration of the possible kinds of ActualMeasurement. Its enumeration literals are:

- InService - Indicates that the ActualProjectMilestone associated with the ActualMilestoneKind is when the configuration goes into service.
- Deployed - Indicates that the ActualProjectMilestone associated with the ActualMilestoneKind is a configuration deployment milestone.
- NoLongerUsed - Indicates that the ActualProjectMilestone associated with the ActualMilestoneKind is when the deployed configuration is no longer used.
- OutOfService - Indicates that the ActualProjectMilestone associated with the ActualMilestoneKind is when the in service configuration goes out of service.
- Other - Indicates that the ActualProjectMilestone associated with the ActualMilestoneKind is not one of the standard ActualMilestoneKinds.

Project

Package: Taxonomy

isAbstract: No

Generalization: PropertySet, Desirer, Block

Extension: Class
Description

An element that describes types of time-limited endeavours that are required to meet one or more Capability needs.

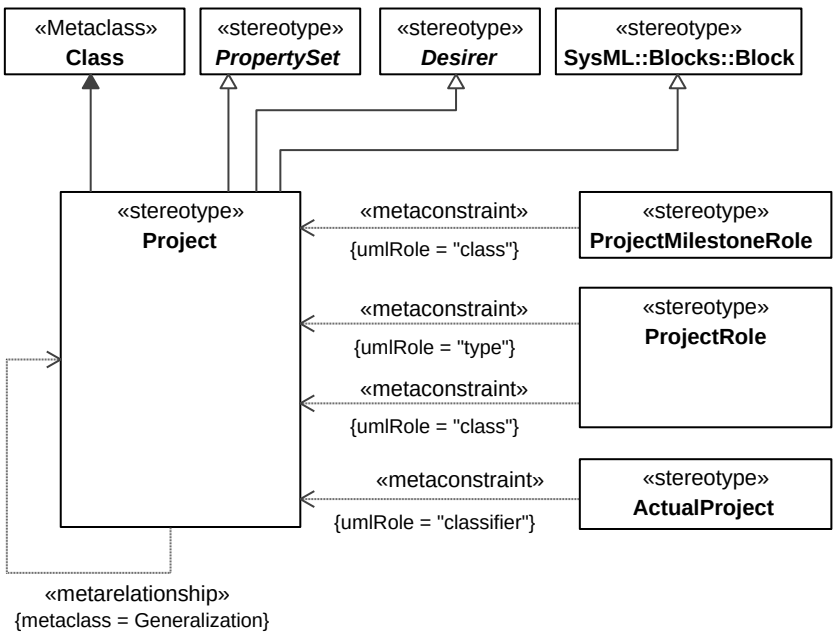

Figure 167 – Project

ProjectKind

Package: Taxonomy

isAbstract: No
Description

Enumeration of the possible kinds of project applicable to an ActualProject. Its enumeration literals are:

- Programme - Indicates that the ActualProject associated with the ProjectKind is an undertaking that is a temporary, flexible organization created to co-ordinate, direct and oversee the implementation of a set of related Projects and Tasks in order to deliver outcomes and benefits related to the organization's strategic objectives. A programme is likely to have a lifespan of several years. During a programme lifecycle, projects are initiated, executed, and closed. Programmes provide an umbrella under which these projects can be co-ordinated. The programme integrates the projects so that it can deliver an outcome greater than the sum of its parts.
- Portfolio - Indicates that the ActualProject associated with the ProjectKind is an undertaking comprised of the Projects and Programmes that are the totality of an organization's investment (or segment thereof) in the changes required to achieve its strategic objectives.
- Project - Indicates that the ActualProject associated with the ProjectKind is an undertaking that is a time-limited endeavor to create a specific set of products or services.
- PersonnelDevelopment - Indicates that the ActualProject associated with the ProjectKind is an undertaking that relates to the training and enablement of personnel to enable them help achieve the organizations objectives.

ProjectMilestone

Package: Taxonomy

isAbstract: No

Generalization: PropertySet, Block

Extension: Class
Description

A type of event in a Project by which progress is measured.

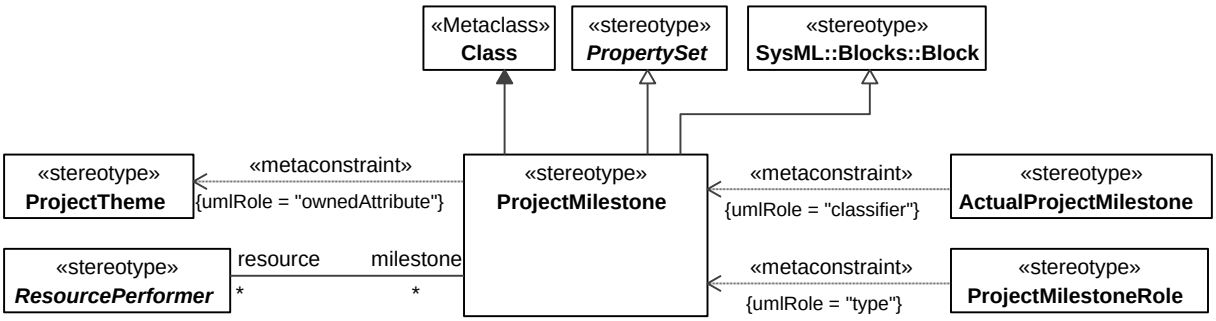

Figure 168 – ProjectMilestone

Associations

resource : ResourcePerformer[*] Relates a ProjectMilestone to the Resources that can be affected by the milestone. It is used to describe aspects of the lifecycle of a Resource.

Constraints

[1] ProjectMilestone.ownedAttribute All of the «ProjectThemes», owned by a «ProjectMilestone», must be typed by the same «StatusIndicators» or its specializations.

7.1.10.2 UAF::Project::Structure

Contains the elements that contribute to the Project Structure Viewpoint.

ProjectMilestoneRole

Package: Structure

isAbstract: No

Generalization: MeasurableElement

Extension: Property
Description

The role played by a ProjectMilestone in the context of a Project.

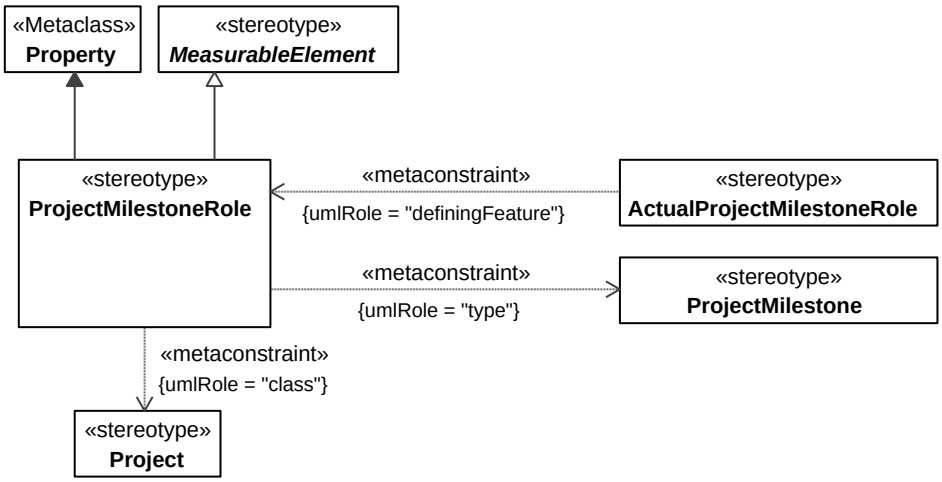

Figure 169 – ProjectMilestoneRole

Constraints

[1] ProjectMilestoneRole.class	Value for the class metaproperty must be stereotyped «Project» or its specializations.
[2] ProjectMilestoneRole.type	Value for the type metaproperty must be stereotyped «ProjectMilestone» or its specializations.

ProjectRole

Package: Structure

isAbstract: No

Generalization: MeasurableElement

Extension: Property
Description

Usage of a Project in the context of another Project. Creates a whole-part relationship.

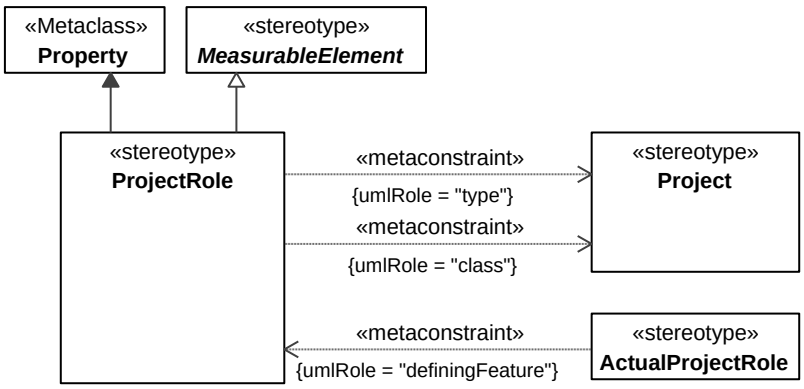

Figure 170 – ProjectRole

Constraints

[1] ProjectRole.class Value for the class metaproperty must be stereotyped «Project» or its specializations.

[2] ProjectRole.type Value for the type metaproperty must be stereotyped «Project» or its specializations.

ProjectStatus

Package: Structure

isAbstract: No

Generalization: UAFElement

Extension: Slot

Description

The status (i.e. level of progress) of a ProjectTheme for an ActualProject at the time of the ActualProjectMilestone.

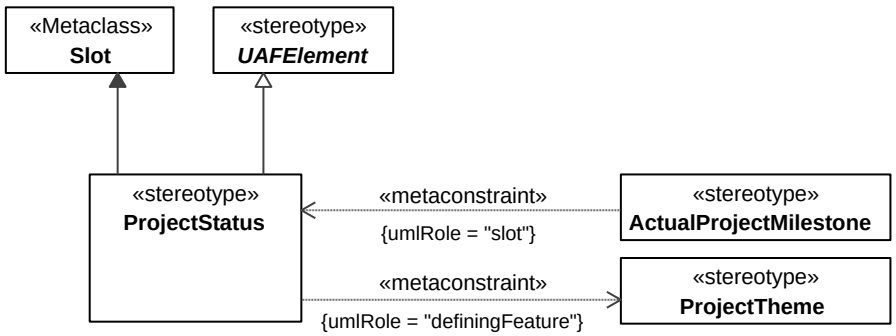

Figure 171 – ProjectStatus

Constraints

[1] ProjectStatus.definingFeature Value for the DefiningFeature metaproperty must be stereotyped «ProjectTheme» or its specializations.

ProjectTheme

Package: Structure

isAbstract: No

Generalization: MeasurableElement

Extension: Property

Description

A property of a ProjectMilestone that captures an aspect by which the progress of ActualProjects may be measured.

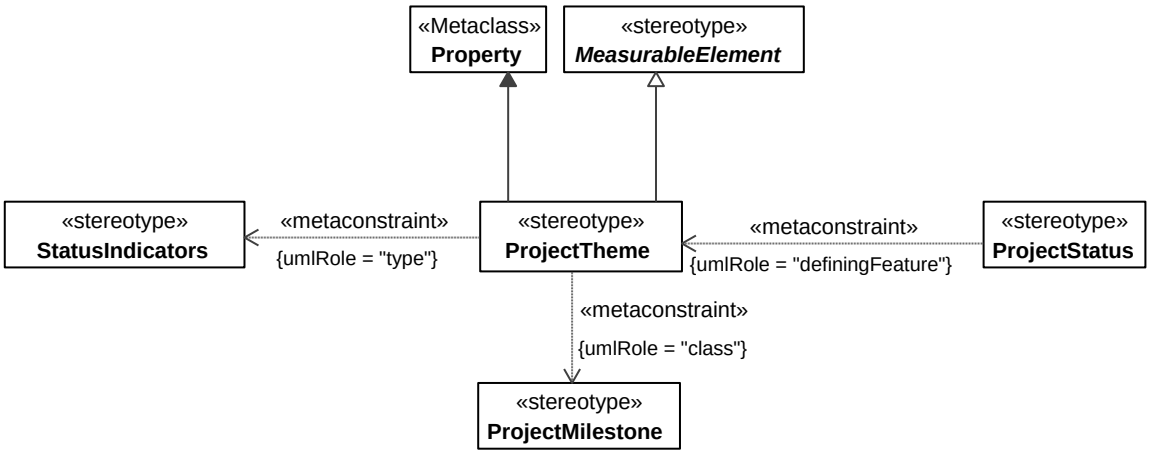

Figure 172 – ProjectTheme

Constraints

[1] ProjecTheme.class Value for the class metaproperty must be stereotyped «ProjectMilestone» or its specializations.

[2] ProjecTheme.type Value for the type metaproperty must be stereotyped «StatusIndicators» or its specializations.

StatusIndicators

Package: Structure

isAbstract: No

Generalization: MeasurableElement, ValueType

Extension: Enumeration

Description

An enumerated type that specifies a status for a ProjectTheme.

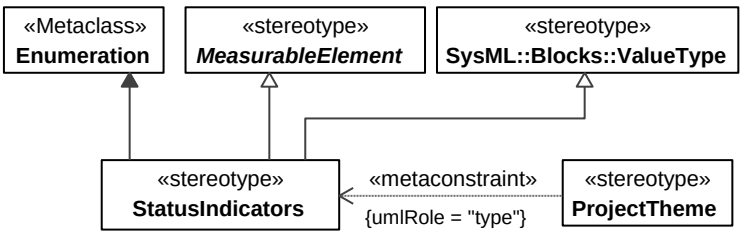

Figure 173 – StatusIndicators

7.1.10.3 UAF::Project::Connectivity

Contains the elements that contribute to the Project Connectivity Viewpoint.

MilestoneDependency

Package: Connectivity

isAbstract: No

Generalization: MeasurableElement

Extension: Dependency

Description

A dependency relationship between two ActualProjectMilestones that denotes one ActualProjectMilestone follows from another.

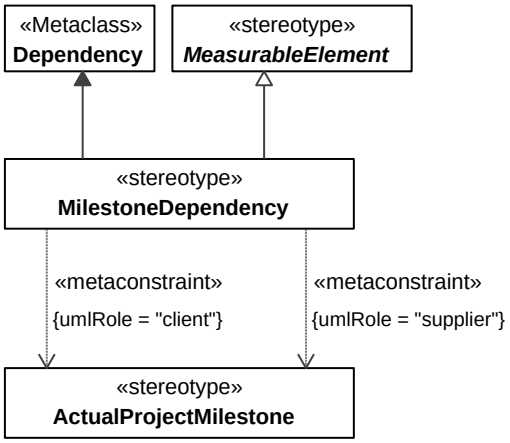

Figure 174 – MilestoneDependency

Constraints

[1] MilestoneDependency.client Value for the client metaproperty must be stereotyped «ActualProjectMilestone» or its

specializations.

[2] MilestoneSequence.supplier	Value for the supplier metaproperty must be stereotyped «ActualProjectMilestone» or its specializations.

ProjectSequence

Package: Connectivity

isAbstract: No

Generalization: MeasurableElement

Extension: Dependency

Description

A dependency relationship between two ActualProjects that denotes one ActualProject cannot start before the previous ActualProject is finished.

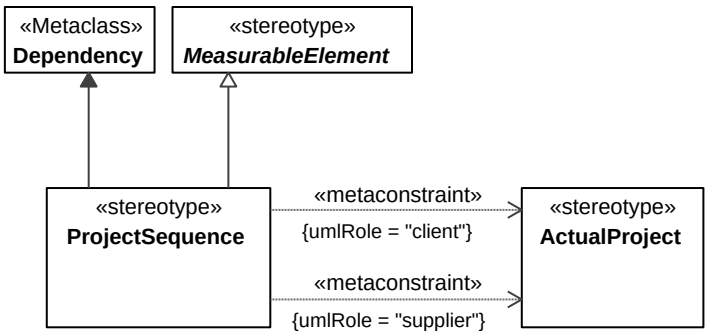

Figure 175 – ProjectSequence

Constraints

[1] ProjectSequence.client	Value for the client metaproperty must be stereotyped «ActualProject» or its specializations.
[2] ProjectSequence.supplier	Value for the supplier metaproperty must be stereotyped «ActualProject» or its specializations.

7.1.10.4 UAF::Project::Roadmap

Contains the elements that contribute to the Project Roadmap Viewpoint.

ActualProject

Package: Roadmap

isAbstract: No

Generalization: ActualPropertySet, Achiever

Extension: InstanceSpecification

Description

A time-limited endeavor to provide a specific set of ActualResources that meet specific Capability needs.

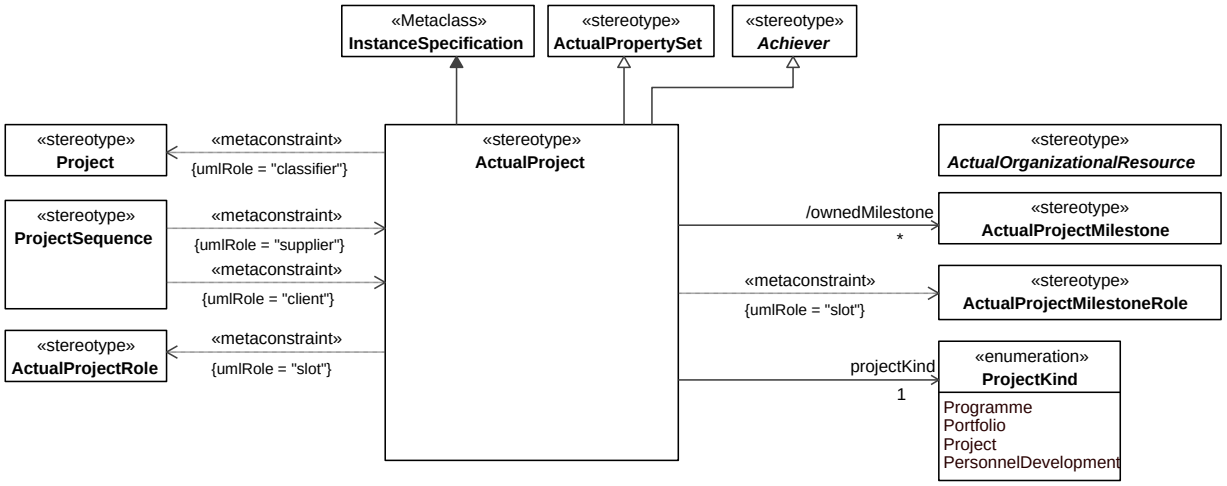

Figure 176 – ActualProject

Associations

ownedMilestone : ActualProjectMilestone[*] Relates the ActualProjectMilestones to the relevant ActualProject.

projectKind : ProjectKind[1] Enumerated value describing the kind of ActualProject.

Constraints

[1] ActualProject.classifier Value for the classifier metaproperty must be stereotyped «Project» or its specializations.

[2] ActualProject.slot Value for the slot metaproperty must be stereotyped «ActualProjectRole», «ActualProjectMilestoneRole», or their specializations.

ActualProjectMilestone

Package: Roadmap

isAbstract: No

Generalization: ActualPropertySet

Extension: InstanceSpecification

Description

An event with a start date in a ActualProject from which progress is measured.

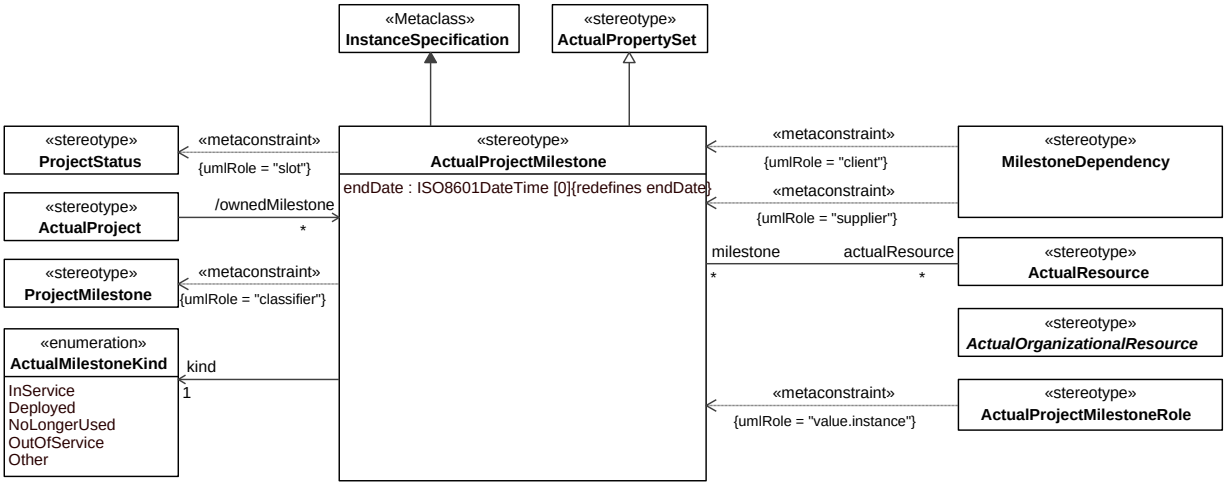

Figure 177 – ActualProjectMilestone

Attributes

endDate : ISO8601DateTime[0] End time for this ActualProjectMilestone.

Associations

actualResource : ActualResource[*]	Relates an ActualProjectMilestone to the ActualResources that are affected by the milestone. It is used to describe aspects of the lifecycle of an ActualResource.
kind : ActualMilestoneKind[1]	Enumerated value describing the kind of ActualProjectMilestone.
versionReleased : VersionedElement[*]	
versionWithdrawn : VersionedElement[*]	

Constraints

[1] ActualProjectMilestone.classifier	Value for the classifier metaproperty must be stereotyped «ProjectMilestone» or its specializations.
[2] ActualProjectMilestone.slot	Value for the slot metaproperty must be stereotyped «ProjectStatus» or its specializations.

ActualProjectMilestoneRole

Package: Roadmap

isAbstract: No

Generalization: ActualState

Extension: Slot

Description

An ActualProjectMilestone that is applied to a ProjectMilestoneRole.

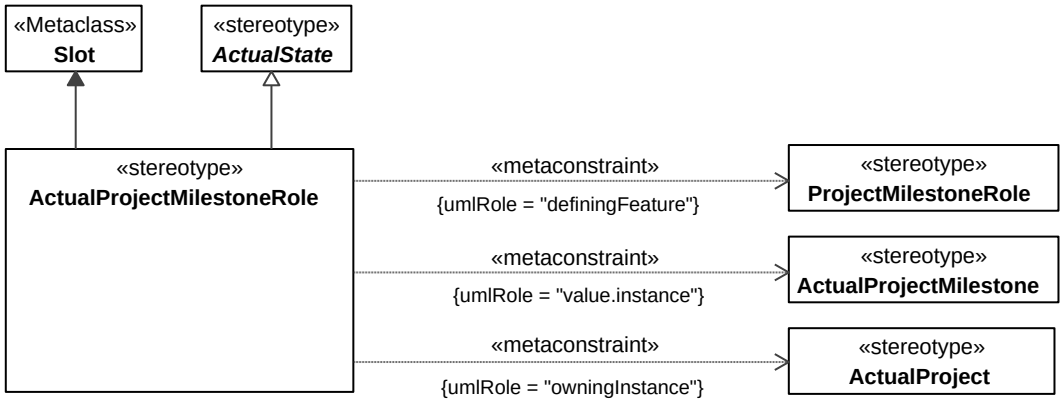

Figure 178 – ActualProjectMilestoneRole

Constraints

[1] ActualProjectMilestoneRole.definingFeature	Value for the definingFeature metaproperty has to be stereotyped «ProjectMilestoneRole» or its specializations.
[2] ActualProjectMilestoneRole.owningInstance	Value for the owningInstance metaproperty has to be stereotyped «ActualProject» or its specializations.
[3] ActualProjectMilestoneRole.value.instance	Value for the value.instance metaproperty has to be stereotyped «ActualProjectMilestone» or its specializations.

ActualProjectRole

Package: Roadmap

isAbstract: No

Generalization: ActualState

Extension: Slot

Description

An ActualProject that is applied to a ProjectRole.

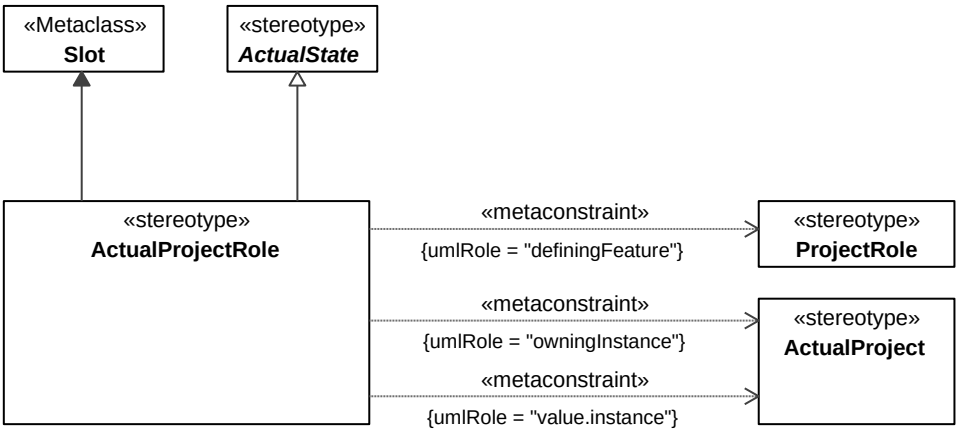

Figure 179 – ActualProjectRole

Constraints

[1] ActualProjectRole.definingFeature	Value for the definingFeature metaproperty has to be stereotyped «ProjectRole» or its specializations.
[2] ActualProjectRole.owningInstance	Value for the owningInstance metaproperty has to be stereotyped «ActualProject» or its specializations.
[3] ActualProjectRole.value.instance	Value for the value.instance metaproperty has to be stereotyped «ActualProject» or its specializations.

7.1.11 UAF::Standards

Stakeholders: Solution Providers, Systems Engineers, Software Engineers, Systems Architects, Business Architects.
Concerns: technical and non-technical Standards applicable to the architecture.
Definition: shows the technical, operational, and business Standards applicable to the architecture. Defines the underlying current and expected Standards.

7.1.11.1 UAF::Standards::Taxonomy

Contains the elements that contribute to the Standards Taxonomy Viewpoint.

Protocol

Package: Taxonomy

isAbstract: No

Generalization: Standard

Extension: Class
Description

A Standard for communication over a network. Protocols may be composite, represented as a ProtocolStack made up of ProtocolLayers.

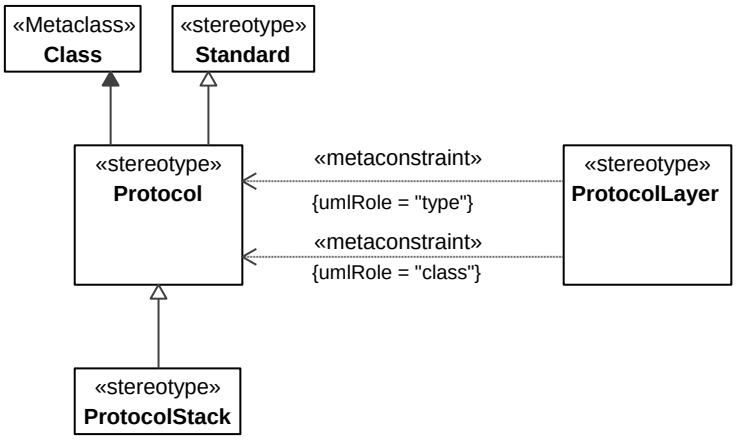

Figure 180 – Protocol

ProtocolStack

Package: Taxonomy

isAbstract: No

Generalization: Protocol

Extension: Class
Description

A sub-type of Protocol that contains the ProtocolLayers, defining a complete stack.

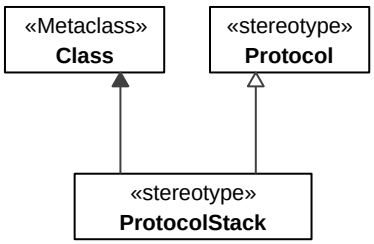

Figure 181 – ProtocolStack

Standard

Package: Taxonomy

isAbstract: No

Generalization: SubjectOfForecast, PropertySet, Block

Extension: Class
Description

A ratified and peer-reviewed specification that is used to guide or constrain the architecture. A Standard may be applied to any element in the architecture.

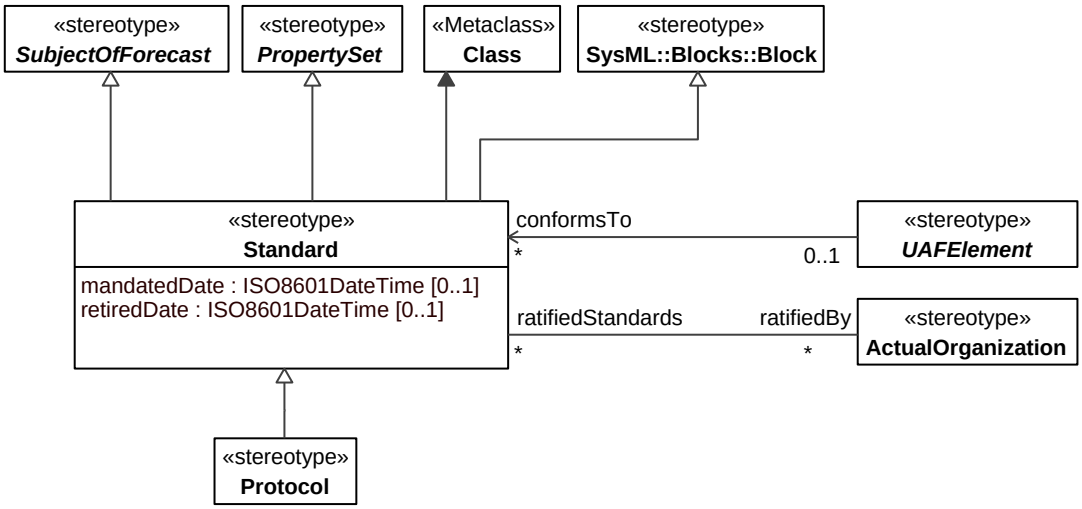

Figure 182 – Standard

Attributes

mandatedDate : ISO8601DateTime[0..1] The date when this version of the Standard was published.

retiredDate : ISO8601DateTime[0..1] The date when this version of the Standard was retired.

Associations

ratifiedBy : ActualOrganization[*] Relates a Standard to the ActualOrganization that ratified the Standard.

7.1.11.2 UAF::Standards::Structure

Contains the elements that contribute to the Standards Structure Viewpoint.

ProtocolLayer

Package: Structure

isAbstract: No

Generalization: MeasurableElement

Extension: Property

Description

Usage of a Protocol in the context of another Protocol. Creates a whole-part relationship.

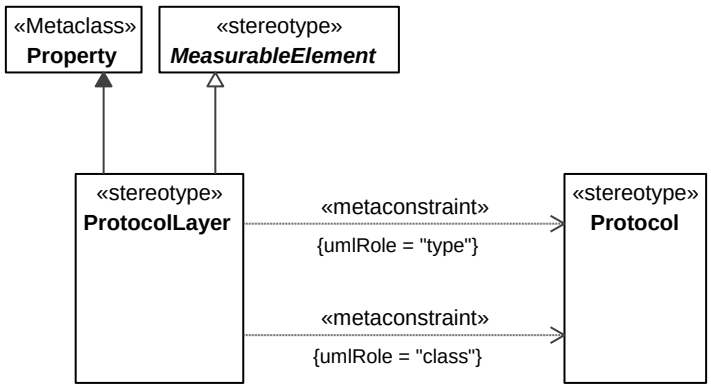

Figure 183 – ProtocolLayer

Constraints

[1] ProtocolLayer.class Value for the class metaproperty must be stereotyped «Protocol» or its specializations.

[2] ProtocolLayer.type Value for the type metaproperty must be stereotyped «Protocol» or its specializations.

7.1.12 UAF::Actual Resources

Stakeholders: Solution Providers, Systems Engineers, Business Architects, Human Resources.
Concerns: the analysis.e.g. evaluation of different alternatives, what-if, trade-offs, V&V on the actual resource configurations.
Definition: illustrates the expected or achieved actual resource configurations and actual relationships between them.

7.1.12.1 UAF::Actual Resources::Taxonomy

Contains the elements that contribute to the Actual Resources Taxonomy Viewpoint.

ActualOrganization

Package: Taxonomy

isAbstract: No

Generalization: ActualResponsibleResource

Extension: InstanceSpecification
Description

An actual formal or informal organizational unit, e.g. "Driving and Vehicle Licensing Agency", "UAF team Alpha".

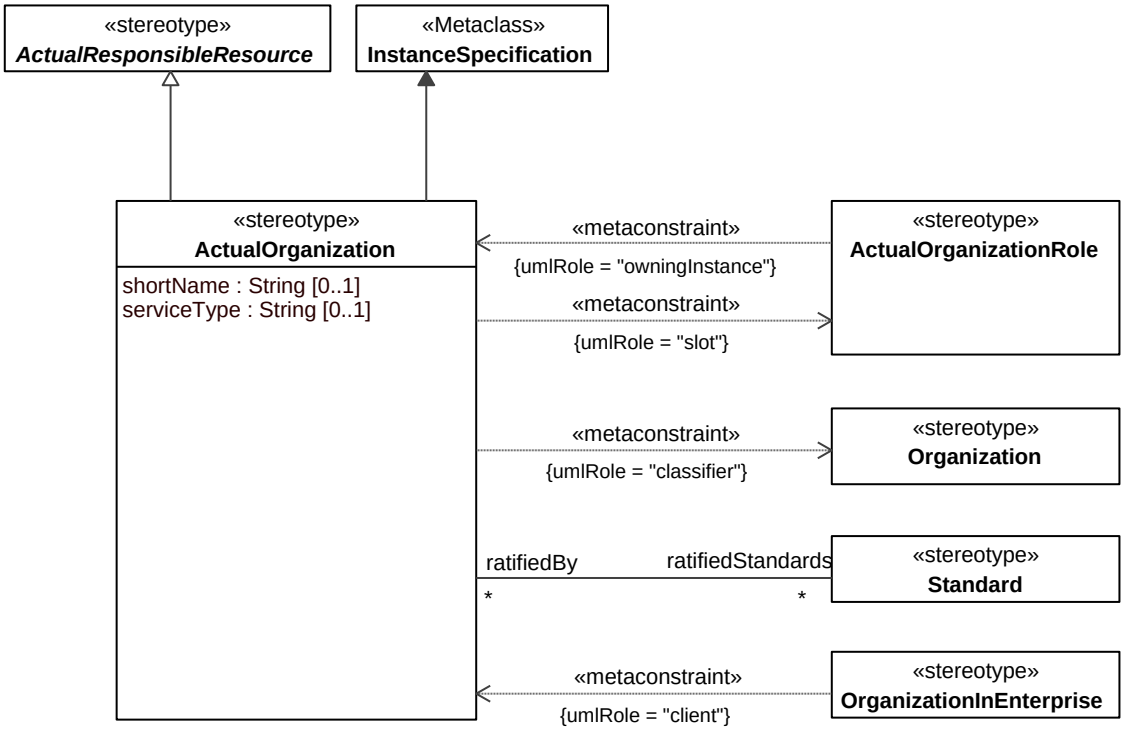

Figure 184 – ActualOrganization

Attributes

serviceType : String[0..1]	Service office code or symbol
shortName : String[0..1]	String providing a simplified means of identifying an ActualOrganization, i.e. SoftWareGroup could use SWG as the shortName.

Associations

ratifiedStandards : Standard[*]	Standards that were ratified by this ActualOrganization.

Constraints

[1] ActualOrganization.classifier	Classifier metaproperty value must be stereotyped «Organization» or its specializations.
[2] ActualOrganization.slot	Slot metaproperty value must be stereotyped «ActualOrganizationRole» or its

specializations.

ActualOrganizationalResource

Package: Taxonomy

isAbstract: Yes

Generalization: Stakeholder, ActualResource

Extension: InstanceSpecification

Description

Abstract element for an ActualOrganization, ActualPerson or ActualPost.

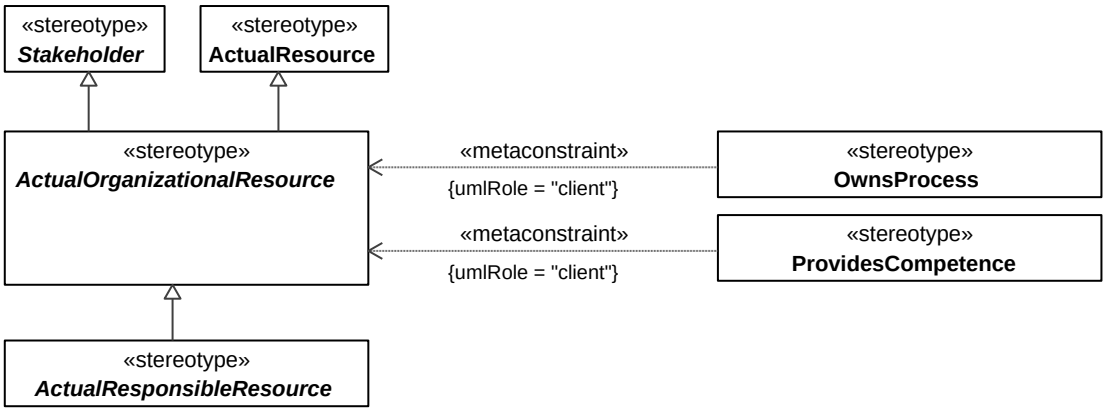

Figure 185 – ActualOrganizationalResource

ActualPerson

Package: Taxonomy

isAbstract: No

Generalization: ActualResponsibleResource

Extension: InstanceSpecification

Description

An individual human being.

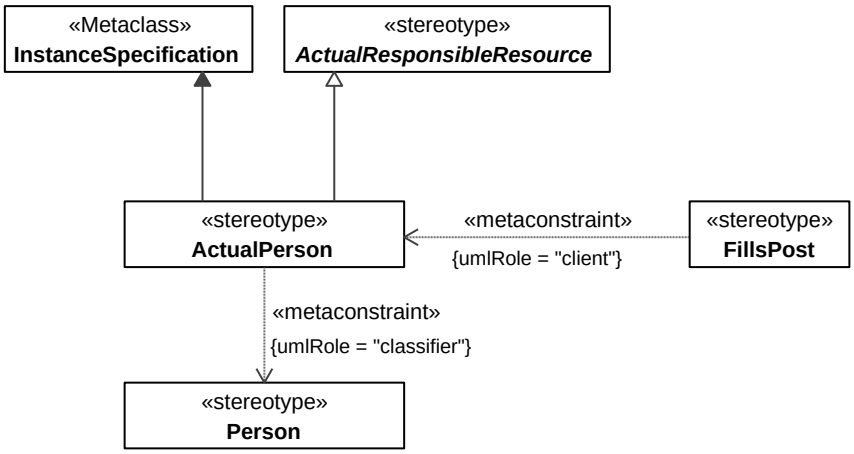

Figure 186 – ActualPerson

Constraints

[1] ActualPerson.classifier Value for the classifier metaproperty has to be stereotyped «Person» or its specializations.

ActualPost

Package: Taxonomy

isAbstract: No

Generalization: ActualResponsibleResource

Extension: InstanceSpecification
Description

An actual, specific post, an instance of a Post "type" - e.g., "President of the United States of America." where the Post would be president.

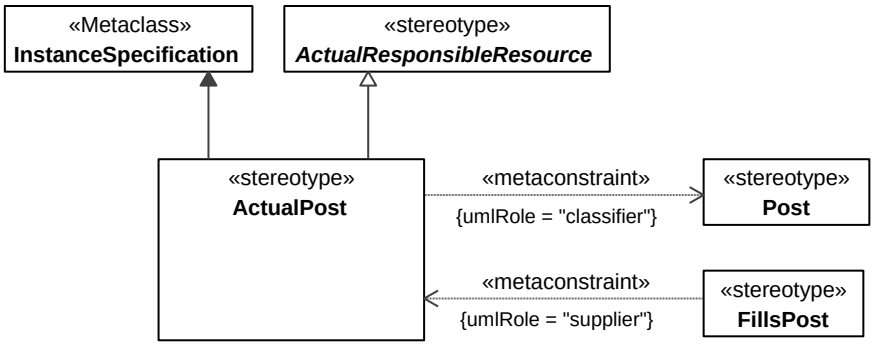

Figure 1:182 - ActualPost

Constraints

[1] ActualPost.classifier Classifier metaproperty value must be stereotyped «Post» or its specializations.

ActualResource

Package: Taxonomy

isAbstract: No

Generalization: ActualPropertySet, SubjectOfResourceConstraint, Achiever

Extension: InstanceSpecification
Description

Role in an Organisation, where the role carries the authority to undertake a function - though the ActualOrganizationalResource given the role has the responsibility.

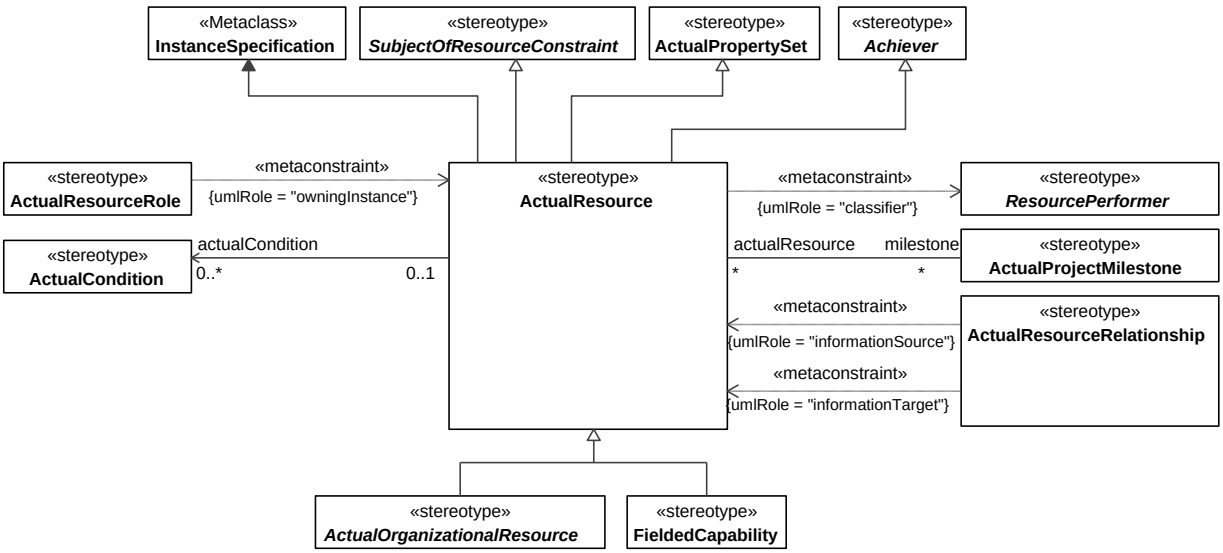

Figure 187 – ActualResource

Associations

actualCondition : ActualCondition[0..*]	Relates the ActualResource to the ActualStates of an environment or location describing its situation
milestone : ActualProjectMilestone[*]	Relates an ActualResource to the ActualProjectMilestones. It is used to describe aspects of the lifecycle of an ActualResource.

Constraints

[1] ActualResource.classifier	Classifier metaproperty value must be stereotyped by a specialization of «ResourcePerformer».

ActualResponsibility

Package: Taxonomy

isAbstract: No

Generalization: ActualOrganizationalResource

Extension: InstanceSpecification

Description

The duty required of a Person or Organization.

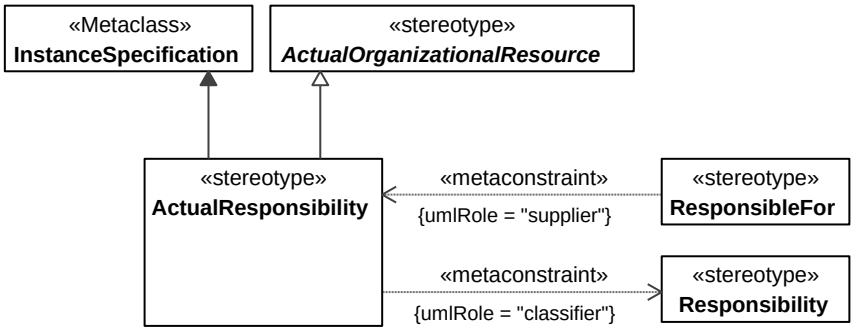

Figure 188 – ActualResponsibility

Constraints

[1] ActualResponsibility.classifier	Classifier metaproperty value must be stereotyped «Responsibility» or its specializations.

ActualResponsibleResource

Package: Taxonomy

isAbstract: Yes

Generalization: ActualOrganizationalResource

Extension: InstanceSpecification

Description

An abstract grouping of responsible OrganizationalResources.

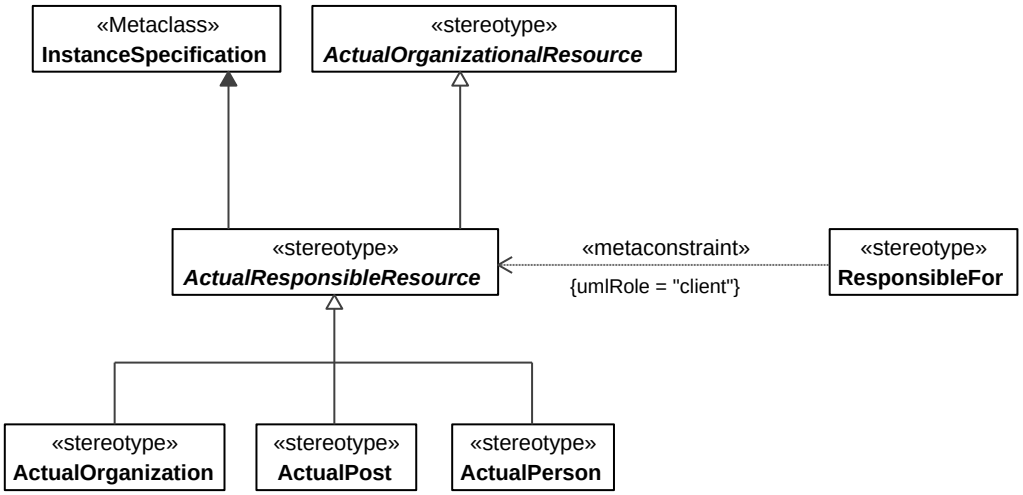

Figure 189 – ActualResponsibleResource

FieldedCapability

Package: Taxonomy

isAbstract: No

Generalization: ActualResource

Extension: InstanceSpecification

Description

An actual, fully-realized capability. A FieldedCapability is typed by a CapabilityConfiguration.

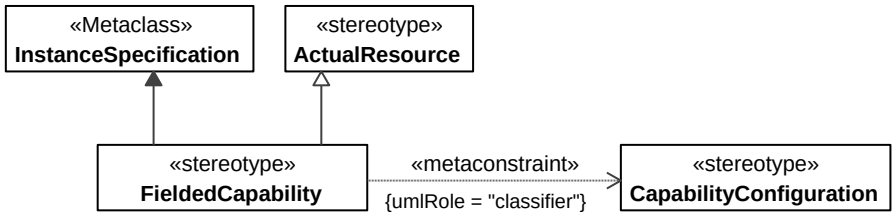

Figure 190 – FieldedCapability

Constraints

[1] FieldedCapability.classifier — Value for the classifier metaproperty must be stereotyped «CapabilityConfiguration» or its specializations.

7.1.12.2 UAF::Actual Resources::Structure

Contains the elements that contribute to the Actual Resources Structure Viewpoint.

ActualOrganizationRole

Package: Structure

isAbstract: No

Generalization: ActualResourceRole

Extension: Slot

Description

An ActualOrganizationalResource that is applied to a ResourceRole.

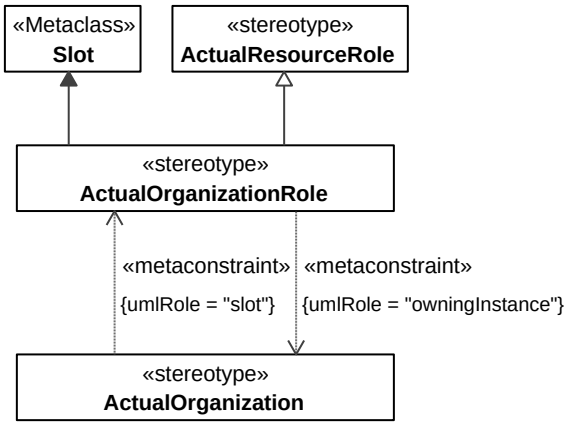

Figure 191 – ActualOrganizationRole

Constraints

[1] ActualOrganizationRole.owningInstance	Value for owningInstance metaproperty has to be stereotyped «ActualOrganization» or its specializations.

ActualResourceRole

Package: Structure

isAbstract: No

Generalization: UAFElement

Extension: Slot

Description

An instance of a ResourcePerformer.

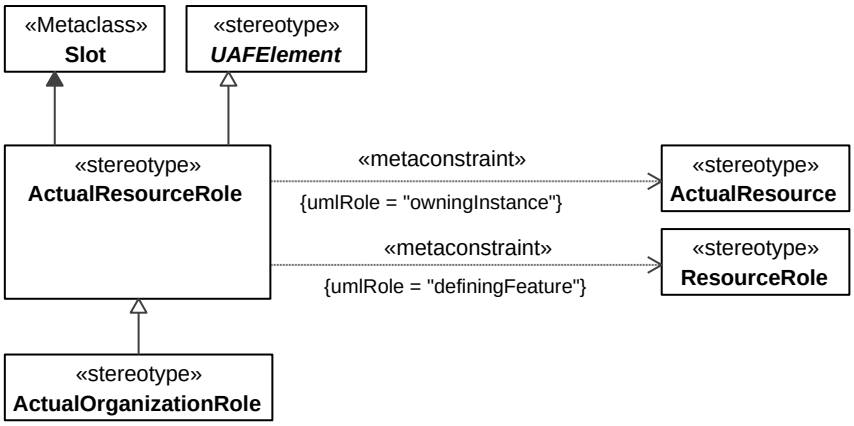

Figure 192 – ActualResourceRole

Constraints

[1] ActualResourceRole.definingFeature	Value for definingFeature metaproperty has to be stereotyped «ResourceRole» or its specializations.
[2] ActualResourceRole.owningInstance	Value for owningInstance metaproperty has to be stereotyped «ActualResource» or its specializations.

7.1.12.3 UAF::Actual Resources::Connectivity

Contains the elements that contribute to the Actual Resources Connectivity Viewpoint.

ActualResourceRelationship

Package: Connectivity

isAbstract: No

Generalization: UAFElement, ItemFlow

Extension: InformationFlow

Description

An abstract element that details the ActualOrganizationalResources that are able to carry out an ActualResponsibility.

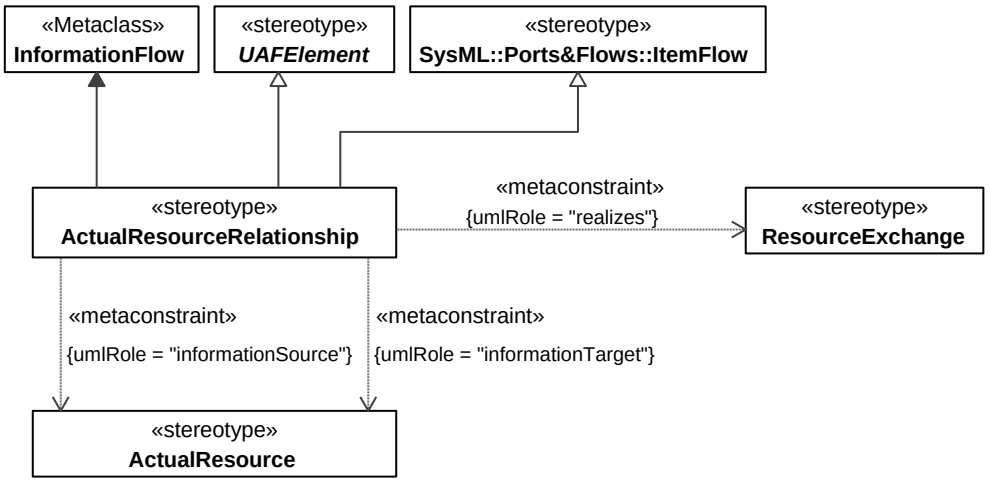

Figure 193 – ActualResourceRelationship

Constraints

[1] ActualResourceRelationship.informationSource	Value for informationSource metaproperty must be stereotyped «ActualResource» or its specializations.
[2] ActualResourceRelationship.informationTarget	Value for informationTarget metaproperty must be stereotyped «ActualResource» or its specializations.
[3] ActualResourceRelationship.realizes	Value for realizes metaproperty must be stereotyped «ResourceExchange» or its specializations.

FillsPost

Package: Connectivity

isAbstract: No

Generalization: MeasurableElement, Allocate

Extension: Abstraction

Description

A dependency relationship that asserts that an ActualPerson fills an ActualPost.

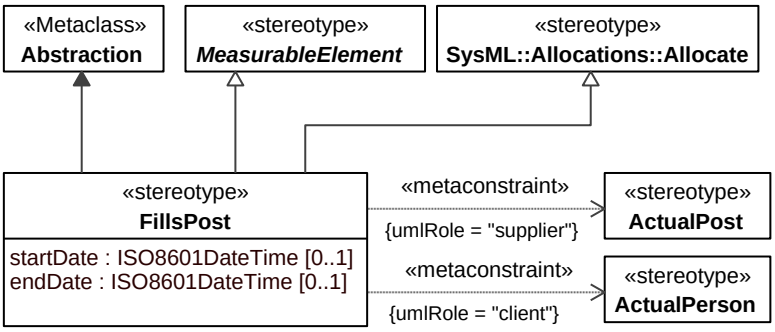

Figure 194 – FillsPost

Attributes

endDate : ISO8601DateTime[0..1] End date of an ActualPerson filling an ActualPost.

startDate : ISO8601DateTime[0..1] Start date of an ActualPerson filling an ActualPost.

Constraints

[1] FillsPost.client Value for the client metaproperty must be stereotyped by «ActualPerson» or its specializations.

[2] FillsPost.supplier Value for the supplier metaproperty must be stereotyped by «ActualPost» or its specializations.

7.1.12.4 UAF::Actual Resources::Constraints

Contains the elements that contribute to the Actual Resources Constraints Viewpoint.

ActualService

Package: Constraints

isAbstract: No

Generalization: ActualMeasurementSet, ActualPropertySet

Extension: InstanceSpecification

Description

An instance of a ServiceSpecification.

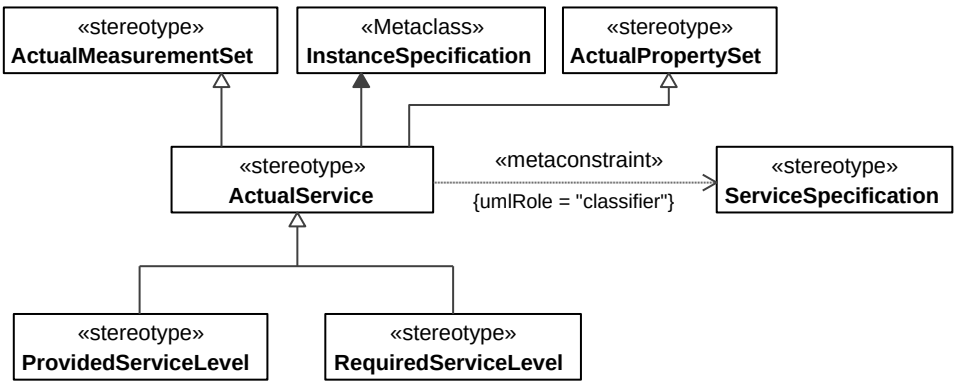

Figure 195 – ActualService

Constraints

[1] ActualService.classifier Value for the classifier metaproperty must be stereotyped by «ServiceSpecification» or its specializations.

ProvidedServiceLevel

Package: Constraints

isAbstract: No

Generalization: ActualService

Extension: InstanceSpecification
Description

A sub type of ActualService that details a specific service level delivered by the provider.

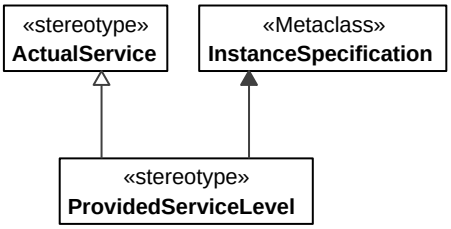

Figure 196 – ProvidedServiceLevel

ProvidesCompetence

Package: Constraints

isAbstract: No

Generalization: MeasurableElement

Extension: Dependency
Description

A dependency relationship that asserts that an ActualOrganizationalResource provides a specific set of Competencies.

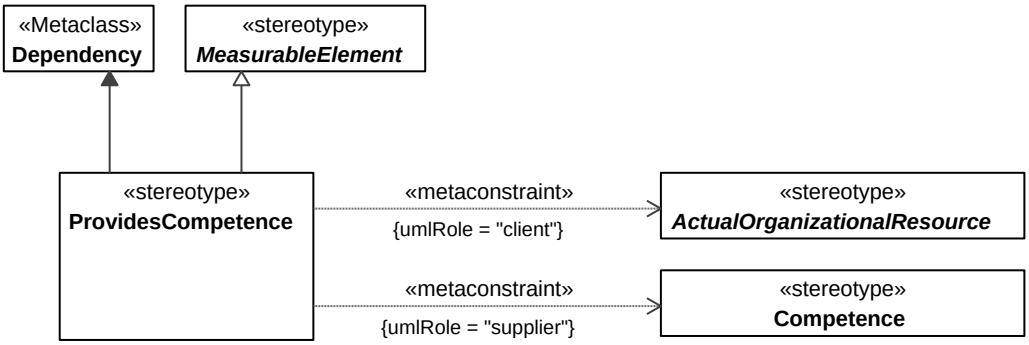

Figure 197 – ProvidesCompetence

Constraints

[1] ProvidesCompetence.client	Value for the client metaproperty must be stereotyped by a specialization of «ActualOrganizationalResource».
[2] ProvidesCompetence.supplier	Value for the supplier metaproperty must be stereotyped «Competence» or its specializations.

RequiredServiceLevel

Package: Constraints

isAbstract: No

Generalization: ActualService

Extension: InstanceSpecification
Description

A sub type of ActualService that details a specific service level required of the provider.

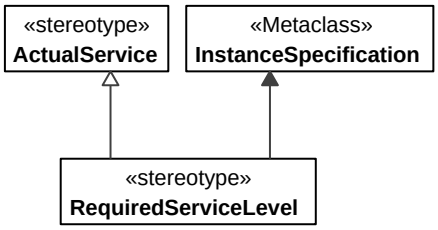

Figure 198 – RequiredServiceLevel

7.1.12.5 UAF::Actual Resources::Traceability

Contains the elements that contribute to the Actual Resources Traceability Viewpoint.

OwnsProcess

Package: Traceability

isAbstract: No

Generalization: MeasurableElement, Allocate

Extension: Abstraction

Description

A dependency relationship denoting that an ActualOrganizationResource owns an OperationalActivity.

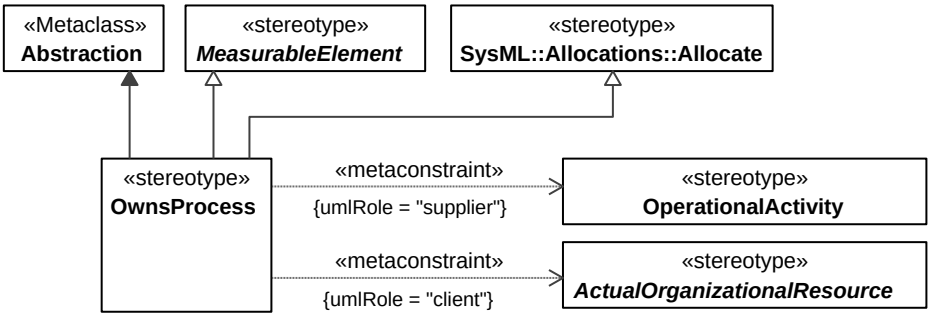

Figure 199 – OwnsProcess

Constraints

[1] OwnsProcess.client	Value for the client metaproperty must be stereotyped «ActualOrganizationalResource» or its specializations.
[2] OwnsProcess.supplier	Value for the supplier metaproperty must be stereotyped «OperationalActivity» or its specializations.

7.1.13 UAF::Summary and Overview

Stakeholders: Executives, PMs, Enterprise Architects.
Concerns: executive-level summary information in a consistent form.
Definition: provides executive-level summary information in a consistent form that allows quick reference and comparison between architectural descriptions. Includes assumptions, constraints, and limitations that may affect high-level decisions relating to an architecture-based work programme.

ArchitecturalDescription

Package: Summary and Overview

isAbstract: No

Generalization: MeasurableElement

Extension: Package

Description

An Architecture Description is a work product used to express the Architecture of some System Of Interest.
It provides executive-level summary information about the architecture description in a consistent form to allow quick reference and comparison between architecture descriptions -- It includes assumptions, constraints, and limitations that may affect high-level decisions relating to an architecture-based work program.

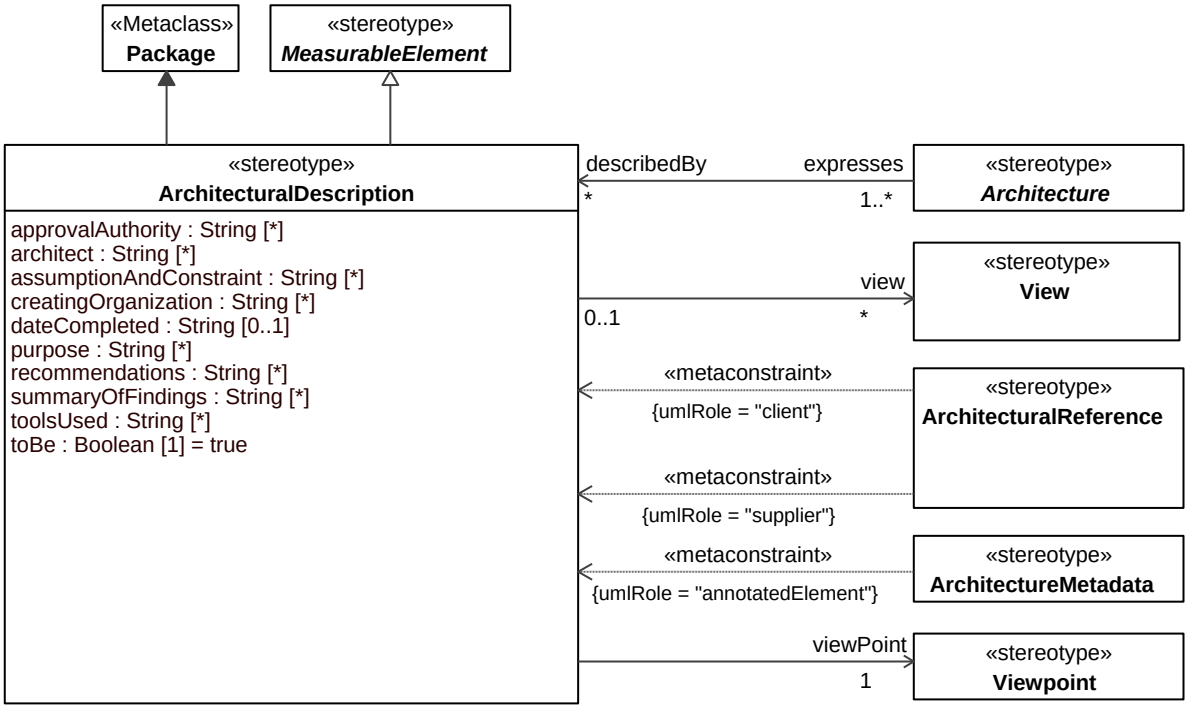

Figure 200 – ArchitecturalDescription

Attributes

approvalAuthority : String[*]	Someone or something that has the authority to approve the ArchitecturalDescription.
architect : String[*]	Someone responsible for the creation of ArchitecturalDescription.
assumptionAndConstraint : String[*]	Any assumptions, constraints, and limitations contained in the ArchitecturalDescription, including those affecting deployment, communications performance, information assurance environments, etc.
creatingOrganization : String[*]	The organization responsible for creating the ArchitecturalDescription.
dateCompleted : String[0..1]	Date that the ArchitecturalDescription was completed.
purpose : String[*]	Explains the need for the Architecture, what it will demonstrate, the types of analyses that will be applied to it, who is expected to perform the analyses, what decisions are expected to be made on the basis of each form of analysis, who is expected to make those decisions, and what actions are expected to result.
recommendations : String[*]	States the recommendations that have been developed based on the architecture effort. Examples include recommended system implementations, and opportunities for technology insertion.
summaryOfFindings : String[*]	Summarizes the findings that have been developed so far. This may be updated several times during the development of the ArchitecturalDescription.
toBe : Boolean[1]	Indicates whether the ArchitecturalDescription represents an Architecture that exists or will exist in the future.
toolsUsed : String[*]	Identifies any tools used to develop the ArchitecturalDescription as well as file names and formats if appropriate.

Associations

architectureFramework : String[1]	Indicates the type of framework used.
view : View[*]	Indicates which views are used in the ArchitecturalDescription.
viewPoint : Viewpoint[1]	Indicates which Viewpoints are used in the ArchitecturalDescription. The definition of Viewpoint corresponds to the definition from ISO/IEC/IEEE 42010.

Architecture

Package: Summary and Overview

isAbstract: Yes

Generalization: UAFElement

Extension: Class

Description

An abstract element that represents a generic architecture. Subtypes are LogicalArchitecture and PhysicalArchitecture.

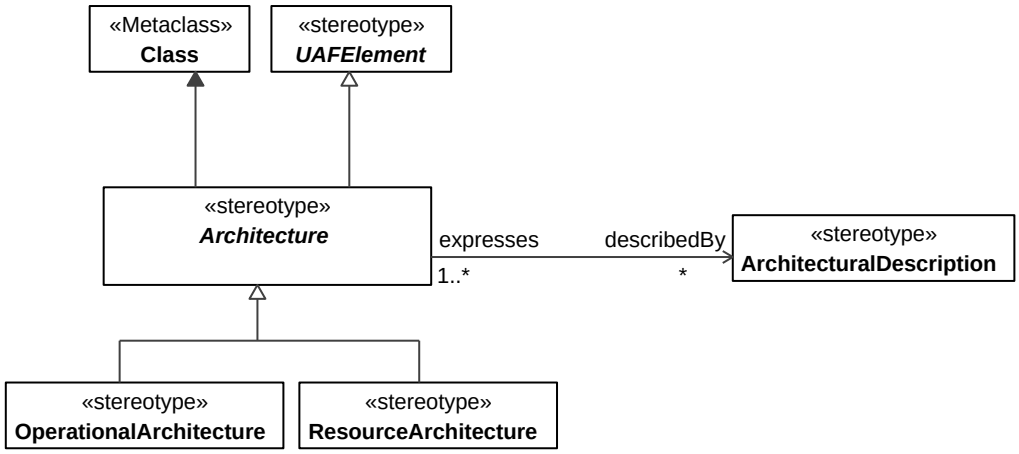

Figure 201 – Architecture

Associations

describedBy : ArchitecturalDescription[*]	The description of an Architecture.

Concern

Package: Summary and Overview

isAbstract: No

Generalization: PropertySet, Block

Extension: Class

Description

Interest in an EnterprisePhase (EnterprisePhase is synonym for System in ISO 42010) relevant to one or more of its stakeholders.

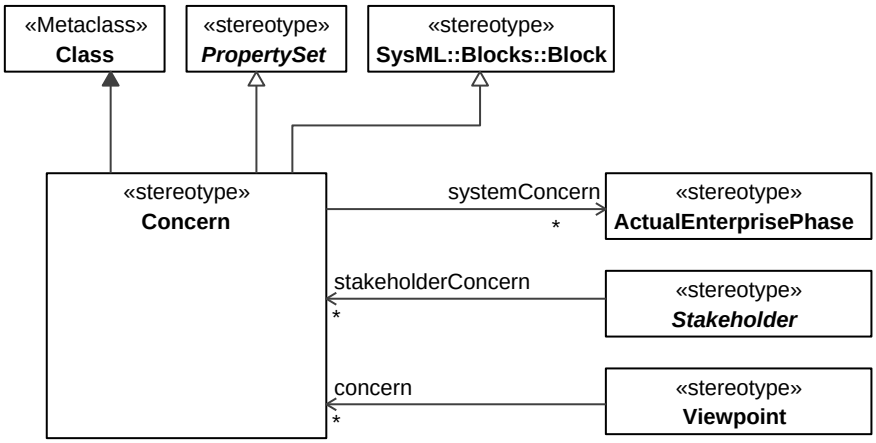

Figure 202 – Concern

Associations

systemConcern : ActualEnterprisePhase[*] Relates a Concern to the ActualEnterprisePhase that addresses that concern (ActualEnterprisePhase is synonym for System in ISO 42010).

Stakeholder

Package: Summary and Overview

isAbstract: Yes

Generalization: UAFElement

Extension: Element

Description

individual, team, organization, or classes thereof, having an interest in an EnterprisePhase [ISO/IEC/IEEE 42010:2011].

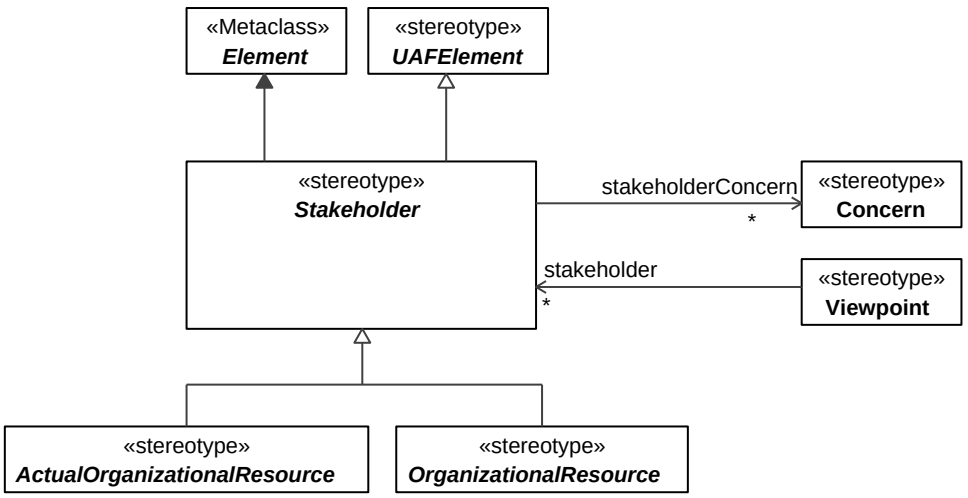

Figure 203 – Stakeholder

Associations

stakeholderConcern : Concern[*] Relates a Stakeholder to a Concern.

UAFElement

Package: Summary and Overview

isAbstract: Yes

Extension: Element

Description

Abstract super type for all of the UAF elements. It provides a way for all of the UAF elements to have a common set of properties.

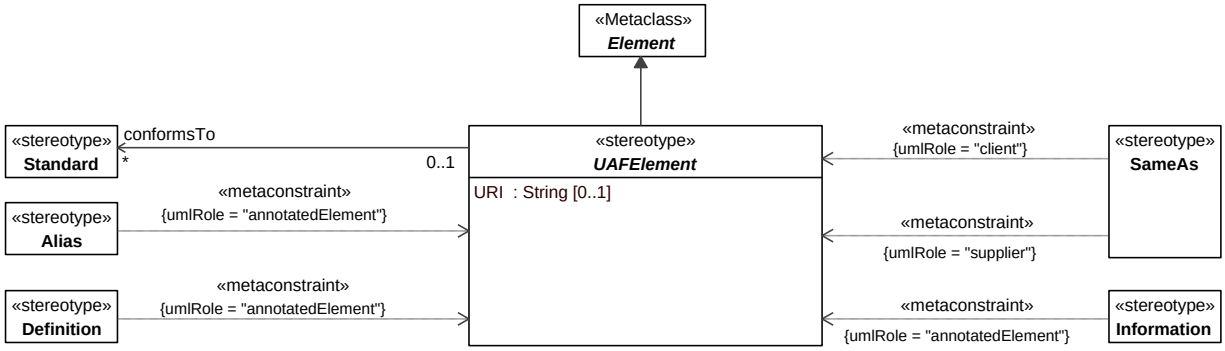

Figure 204 – UAFElement

Attributes

URI : String[0..1] Captures Unique identifier for the element.

Associations

conformsTo : Standard[*] Relates a UAFElement to the Standard that the UAFElement is conforming to.

View

Package: Summary and Overview

isAbstract: No

Generalization: PropertySet, View

Extension: Class

Description

An architecture view expresses the architecture of the system-of-interest in accordance with an architecture viewpoint (or simply, viewpoint). [ISO/IEC/IEEE 42010:2011(E)].

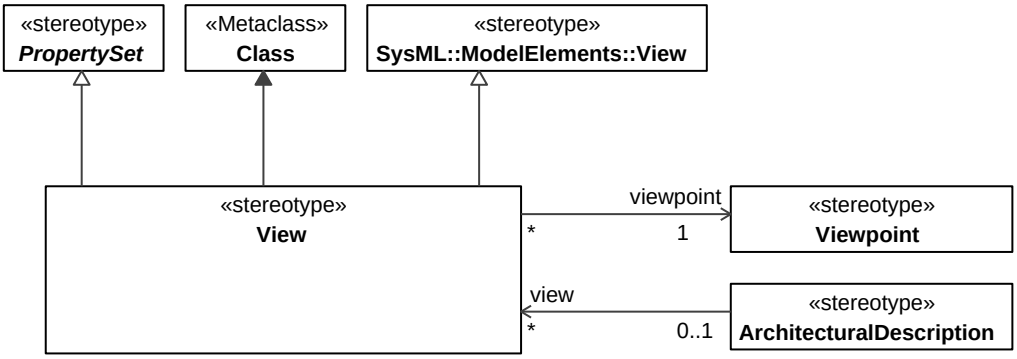

Figure 205 – View

Associations

viewpoint : Viewpoint[1] Relates the View to the Viewpoint that the View conforms to.

Viewpoint

Package: Summary and Overview

isAbstract: No

Generalization: PropertySet, Viewpoint

Extension: Class

Description

An architecture viewpoint frames (to formulate or construct in a particular style or language) one or more concerns. A concern can be framed by more than one viewpoint. [ISO/IEC/IEEE 42010:2011(E)].

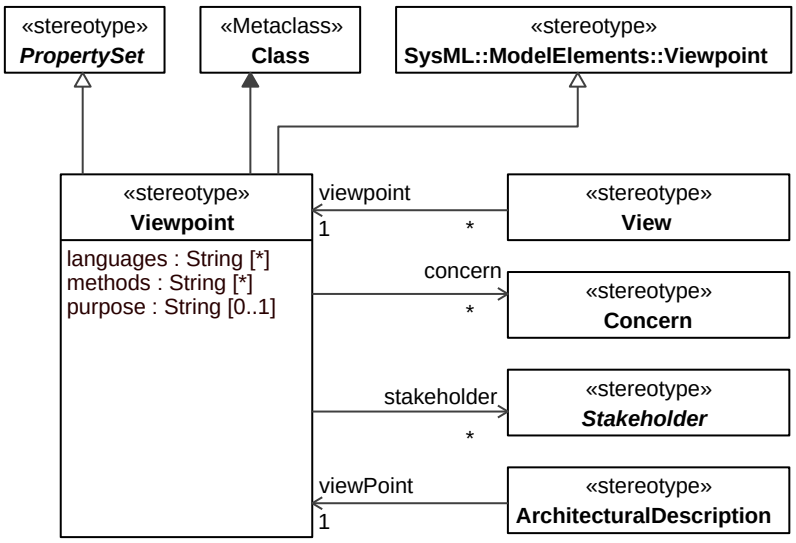

Figure 206 – Viewpoint

Attributes

languages : String[*]	The languages used to express the Viewpoint.
methods : String[*]	The methods employed in the development of the Viewpoint.
purpose : String[0..1]	The purpose of the Viewpoint.

Associations

concern : Concern[*]	Relates the Viewpoint to the Concerns that the Viewpoint addresses.
stakeholder : Stakeholder[*]	Relates the Viewpoint to the Stakeholders whose Concerns are being addressed by the Viewpoint.

Annex A UAFP Views (Profile)

This section is intended as non-normative guidance for developers and users as to what UAFP elements and relationships are applicable for each of the UAFP Views.

View Specifications

MODAF: A connected and coherent set of Architectural Elements which conform to a View
DoDAF Alias: View: DoDAF divides the problem space into manageable pieces, according to the stakeholder's Viewpoint, further defined in the framework as "Views."

View Specifications::Strategic

Stakeholders: Capability Portfolio Managers.
Concerns: capability management process.
Definition: describe capability taxonomy, composition, dependencies and evolution.

View Specifications::Strategic::Taxonomy

Stakeholders: PMs, Enterprise Architects, Executives.
Concerns: capability needs.
Definition: shows the taxonomy of capabilities.
Recommended Implementation: SysML Block Definition Diagram.

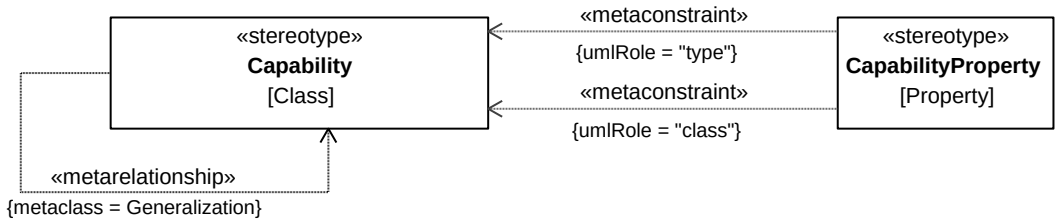

Figure 207 – Strategic Taxonomy

Elements

- Capability
- CapabilityProperty

View Specifications::Strategic::Structure

Stakeholders: PMs, Enterprise Architects, Executives.
Concerns: capability needs.
Definition: shows the composition of capabilities and how capabilities are decomposed from higher level (e.g. Joint capabilities) ones as needed.
Recommended Implementation: SysML Block Definition Diagram.

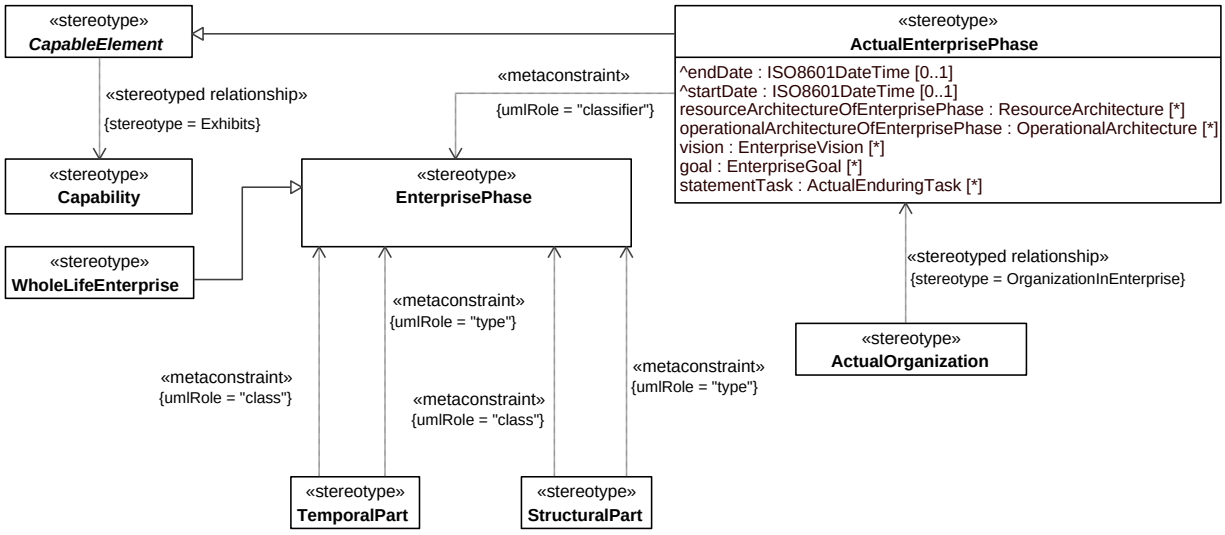

Figure 208 – Strategic Structure

Elements

- ActualEnterprisePhase
- ActualOrganization
- Capability
- CapableElement
- EnterprisePhase
- StructuralPart
- TemporalPart
- WholeLifeEnterprise

View Specifications::Strategic::Connectivity

Stakeholders: PMs, Executives, Enterprise Architects.
Concerns: capability dependencies.
Definition: describes the dependencies between planned capabilities.
Recommended Implementation: SysML Block Definition Diagram. SysML Internal Block Diagram.

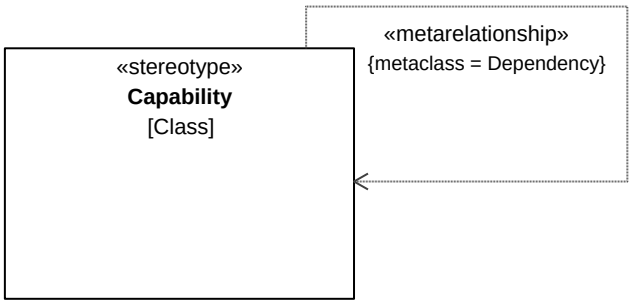

Figure 209 – Strategic Connectivity

Elements

- Capability

View Specifications::Strategic::States

Stakeholders: PMs, Enterprise Architects.
Concerns: effects that the implementation(s) of capabilities are expected to deliver.
Definition: captures the relationships between capability(ies) and desired effect(s) that implementation(s) of capability(ies) should achieve.
Recommended Implementation: SysML Block Definition Diagram.

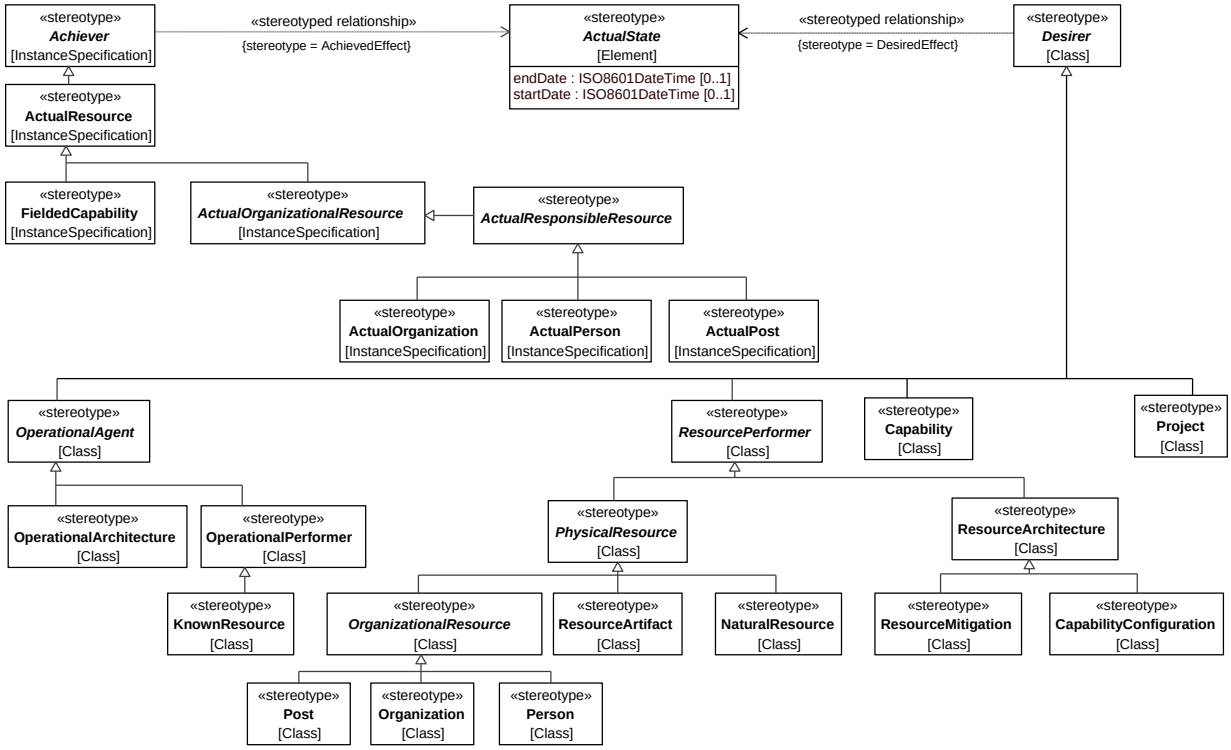

Figure 210 – Strategic States

Elements

- Achiever
- ActualOrganization
- ActualOrganizationalResource
- ActualPerson
- ActualPost
- ActualResource
- ActualResponsibleResource
- ActualState
- Capability
- CapabilityConfiguration
- Desirer
- FieldedCapability
- KnownResource
- NaturalResource
- OperationalAgent
- OperationalArchitecture
- OperationalPerformer
- Organization
- OrganizationalResource
- Person
- PhysicalResource
- Post
- Project
- ResourceArchitecture
- ResourceArtifact
- ResourceMitigation
- ResourcePerformer

View Specifications::Strategic::Constraints

Stakeholders: PMs, Enterprise Architects.
Concerns: capability constraints.
Definition: details the measurements that set performance requirements constraining capabilities.
Recommended Implementation: tabular format, SysML Block Definition Diagram.

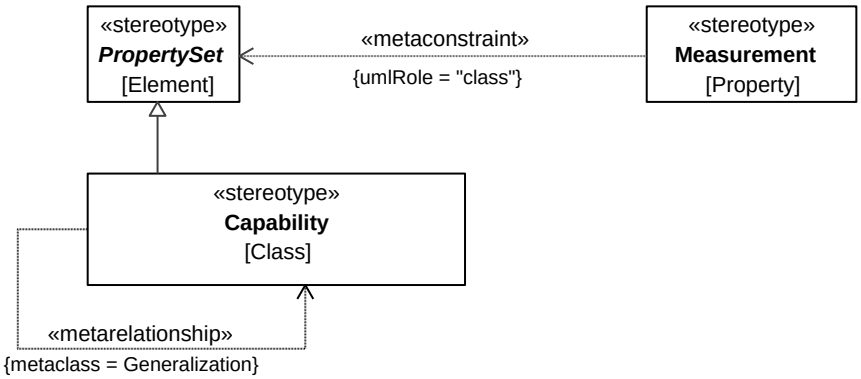

Figure 211 – Strategic Constraints

Elements

- Capability
- Measurement
- PropertySet

View Specifications::Strategic::Roadmap::Strategic Roadmap: Deployment

Stakeholders: PMs, Executives, Enterprise Architects.
Concerns: capability deployment to organizations over time.
Definition: addresses the deployment of capability(ies) to actual organizations over time.
Recommended Implementation: timeline, tabular format, SysML Block Definition Diagram.

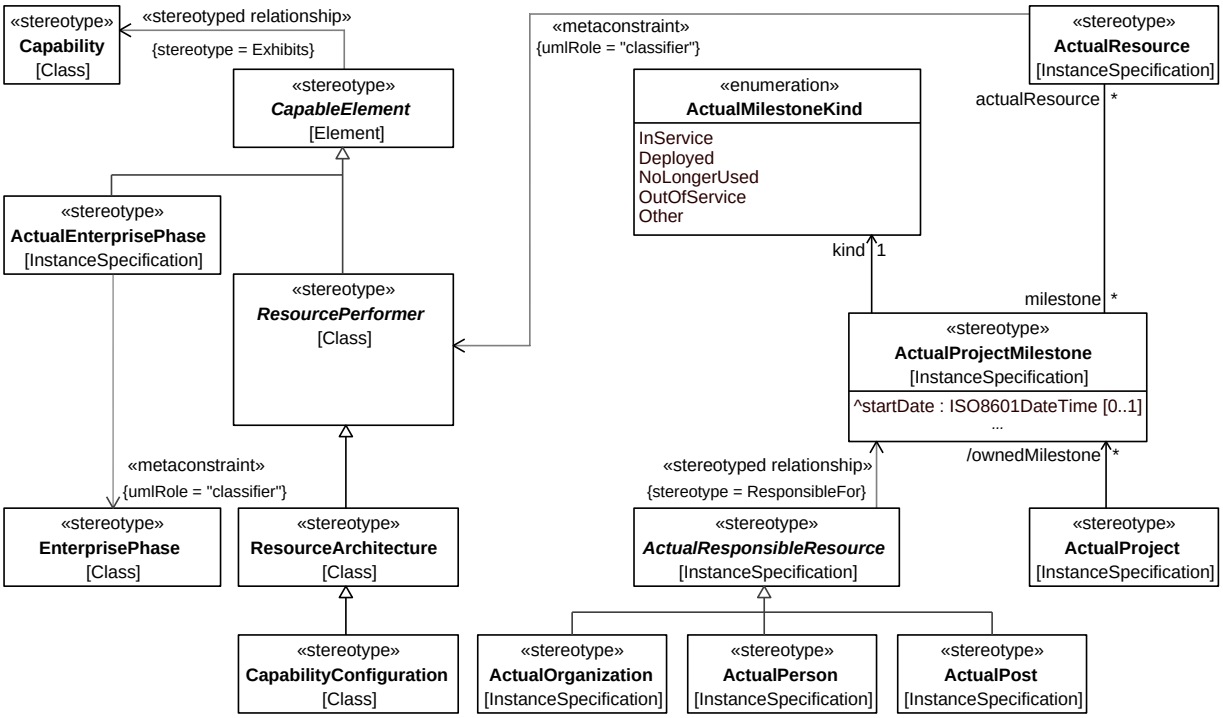

Figure 212 – Strategic Roadmap: Deployment

Elements

- ActualEnterprisePhase
- ActualMilestoneKind
- ActualOrganization
- ActualPerson
- ActualPost
- ActualProject
- ActualProjectMilestone
- ActualResource
- ActualResponsibleResource
- Capability
- CapabilityConfiguration
- CapableElement
- EnterprisePhase
- ResourceArchitecture
- ResourcePerformer

View Specifications::Strategic::Roadmap::Strategic Roadmap: Phasing

Stakeholders: PMs, Executives, Enterprise Architects.
Concerns: capability(ies) achievement over time.
Definition: the planned achievement of capability(ies) at different points in time or during specific periods of time.
Recommended Implementation: timeline, tabular format, SysML Block Definition Diagram.

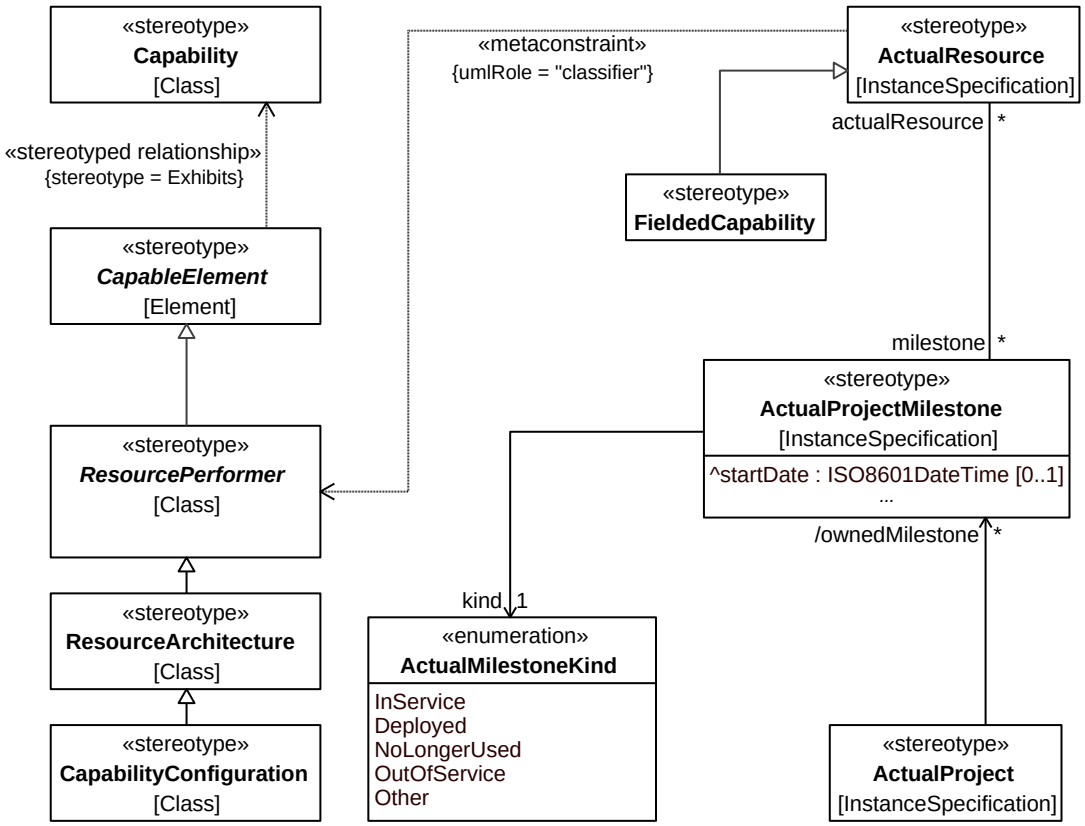

Figure 213 – Strategic Roadmap: Phasing

Elements

- ActualMilestoneKind
- ActualProject
- ActualProjectMilestone
- ActualResource

- Capability
- CapabilityConfiguration
- CapableElement
- FieldedCapability
- ResourceArchitecture
- ResourcePerformer

View Specifications::Strategic::Traceability

Stakeholders: PMs, Enterprise Architects, Business Architects.
Concerns: traceability between capabilities and operational activities.
Definition: describes the mapping between the capabilities required by an Enterprise and the supporting operational activities.
Recommended Implementation: matrix format, SysML Block Definition Diagram.

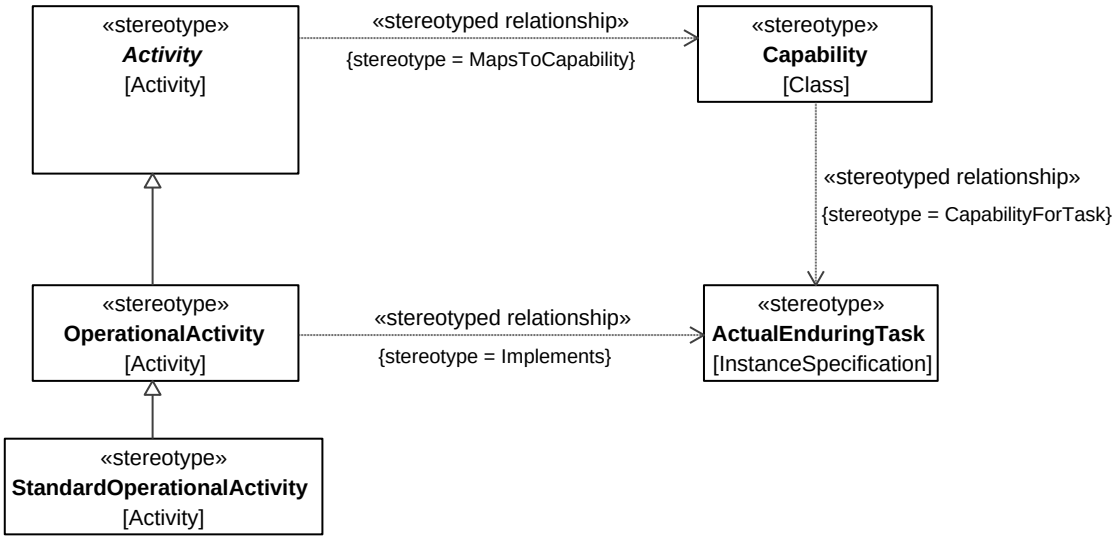

Figure 214 – Strategic Traceability

Elements

- Activity
- ActualEnduringTask
- Capability
- OperationalActivity
- StandardOperationalActivity

View Specifications::Operational

Stakeholders: Business Architects, Executives
Concerns: illustrate the Logical Architecture of the enterprise.
Definition: describe the requirements, operational behavior, structure, and exchanges required to support (exhibit) capabilities. Defines all operational elements in an implementation/solution independent manner.

View Specifications::Operational::Taxonomy

Stakeholders: Business Architects, Systems Engineers, Enterprise Architects, OperationalPerformer Owners.
Concerns: OperationalAgent types.
Definition: shows the taxonomy of types of OperationalAgents.
Recommended Implementation: SysML Block Definition Diagram.

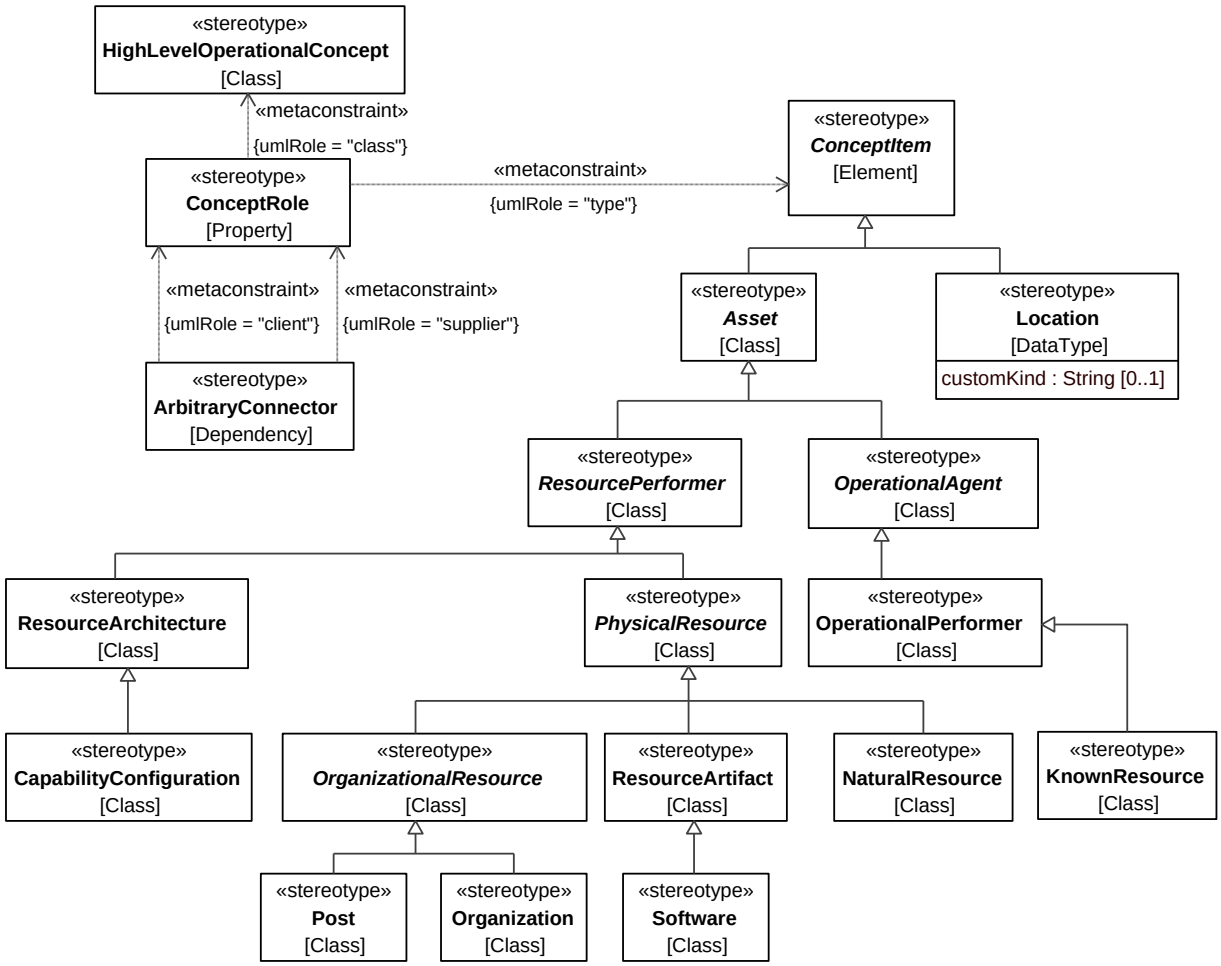

Figure 215 – Operational Taxonomy

Elements

- ArbitraryConnector
- Asset
- CapabilityConfiguration
- ConceptItem
- ConceptRole
- HighLevelOperationalConcept
- KnownResource
- Location
- NaturalResource
- OperationalAgent
- OperationalPerformer
- Organization
- OrganizationalResource
- PhysicalResource
- Post
- ResourceArchitecture
- ResourceArtifact
- ResourcePerformer
- Software

View Specifications::Operational::Structure

Stakeholders: Business Architects, Systems Engineers, Enterprise Architects, OperationalPerformer Owners.
Concerns: identifies the operational exchange requirements between nodes.
Definition: defines operational architecture and exchange requirements necessary to support a specific set of Capability(ies).
Recommended Implementation: SysML Block Definition Diagram, SysML Internal Block Diagram.

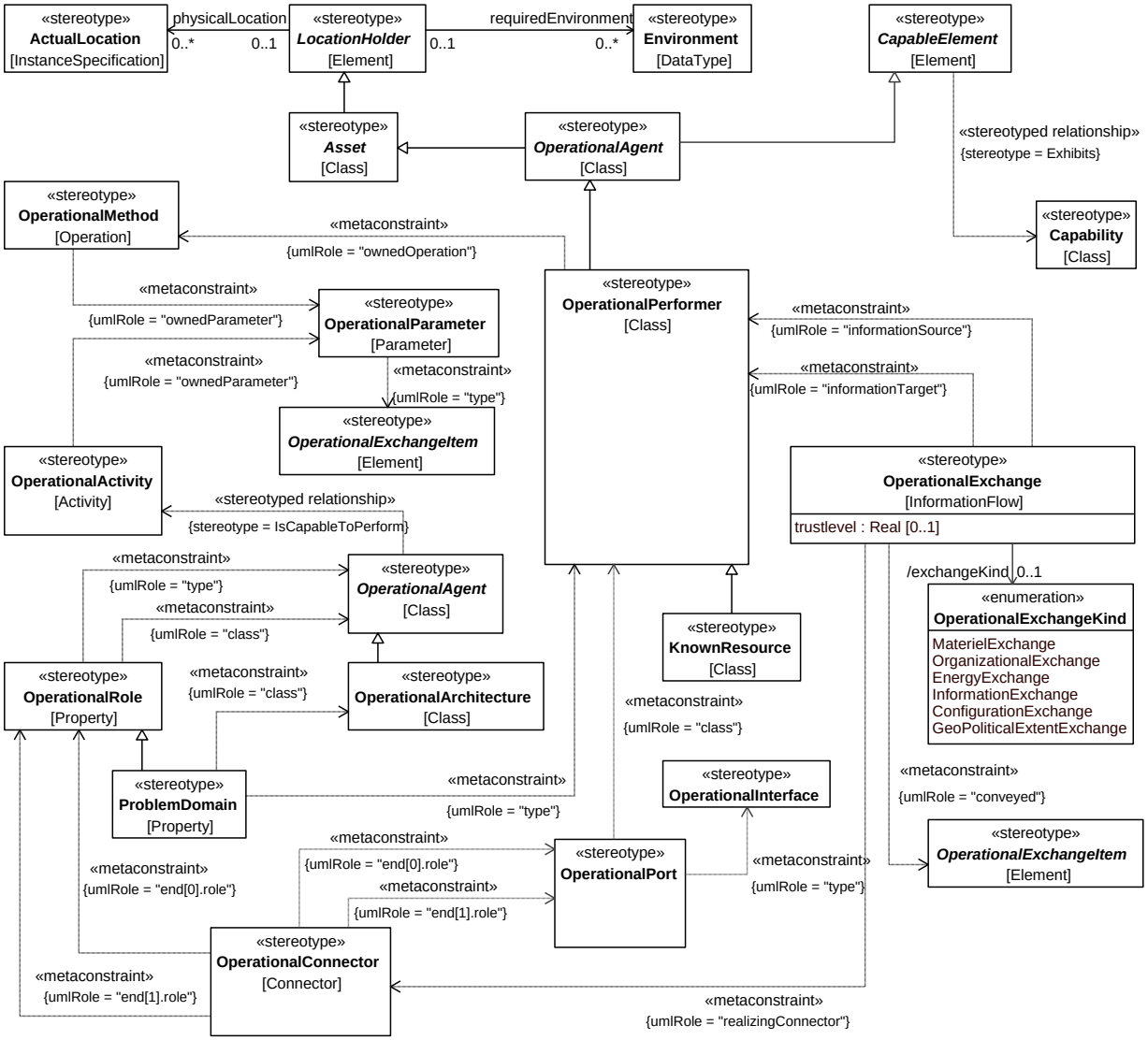

Figure 216 – Operational Structure

Elements

- ActualLocation
- Asset
- Capability
- CapableElement
- Environment
- KnownResource
- LocationHolder
- OperationalActivity
- OperationalAgent
- OperationalArchitecture

- OperationalConnector
- OperationalExchange
- OperationalExchangeItem
- OperationalExchangeKind
- OperationalInterface
- OperationalMethod
- OperationalParameter
- OperationalPerformer
- OperationalPort
- OperationalRole
- ProblemDomain

View Specifications::Operational::Connectivity

Stakeholders: Systems Engineers, Architects, Solution Providers.
Concerns: capture the interfaces between logical nodes.
Definition: summarizes logical exchanges between nodes of information, systems, personnel, energy etc. and the logical activities that produce and consume them. Measurements can optionally be included.
Recommended Implementation: tabular format.

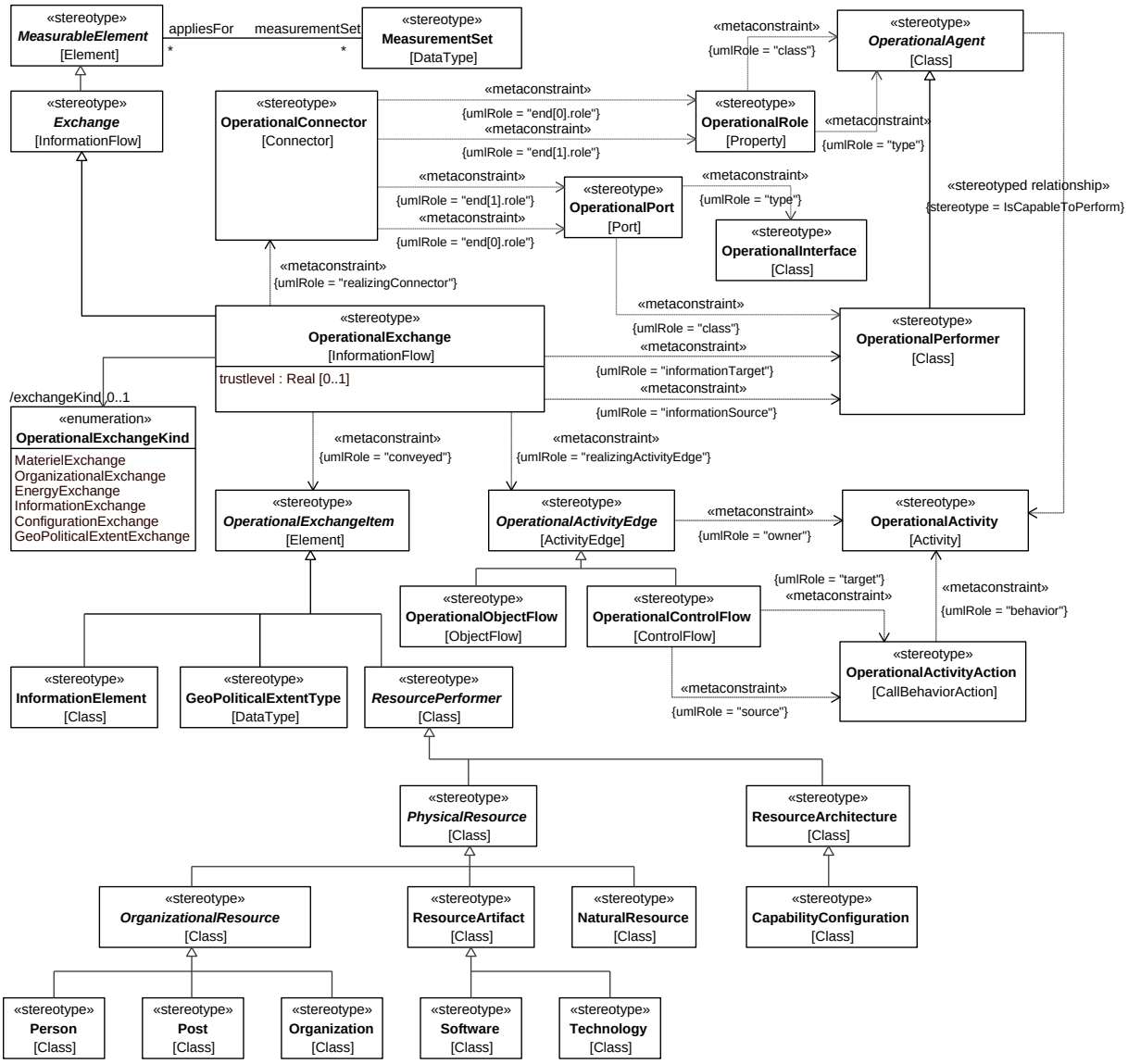

Figure 217 – Operational Connectivity

Elements

- CapabilityConfiguration
- Exchange
- GeoPoliticalExtentType
- InformationElement
- MeasurableElement
- MeasurementSet
- NaturalResource
- OperationalActivity
- OperationalActivityAction
- OperationalActivityEdge
- OperationalAgent
- OperationalConnector
- OperationalControlFlow
- OperationalExchange
- OperationalExchangeItem

- OperationalExchangeKind
- OperationalInterface
- OperationalObjectFlow
- OperationalPerformer
- OperationalPort
- OperationalRole
- Organization
- OrganizationalResource
- Person
- PhysicalResource
- Post
- ResourceArchitecture
- ResourceArtifact
- ResourcePerformer
- Software
- Technology

View Specifications::Operational::Processes

Stakeholders: Business Architect, Systems Engineers, Enterprise Architects
Concerns: captures activity based behavior and flows.
Definition: describes the activities that are normally conducted in the course of achieving business goals that support a capability. It describes operational activities, their Inputs/Outputs, operational activity actions and flows between them.
Recommended Implementation: SysML Activity Diagram, SysML Block Definition Diagram, BPMN Process Diagram.

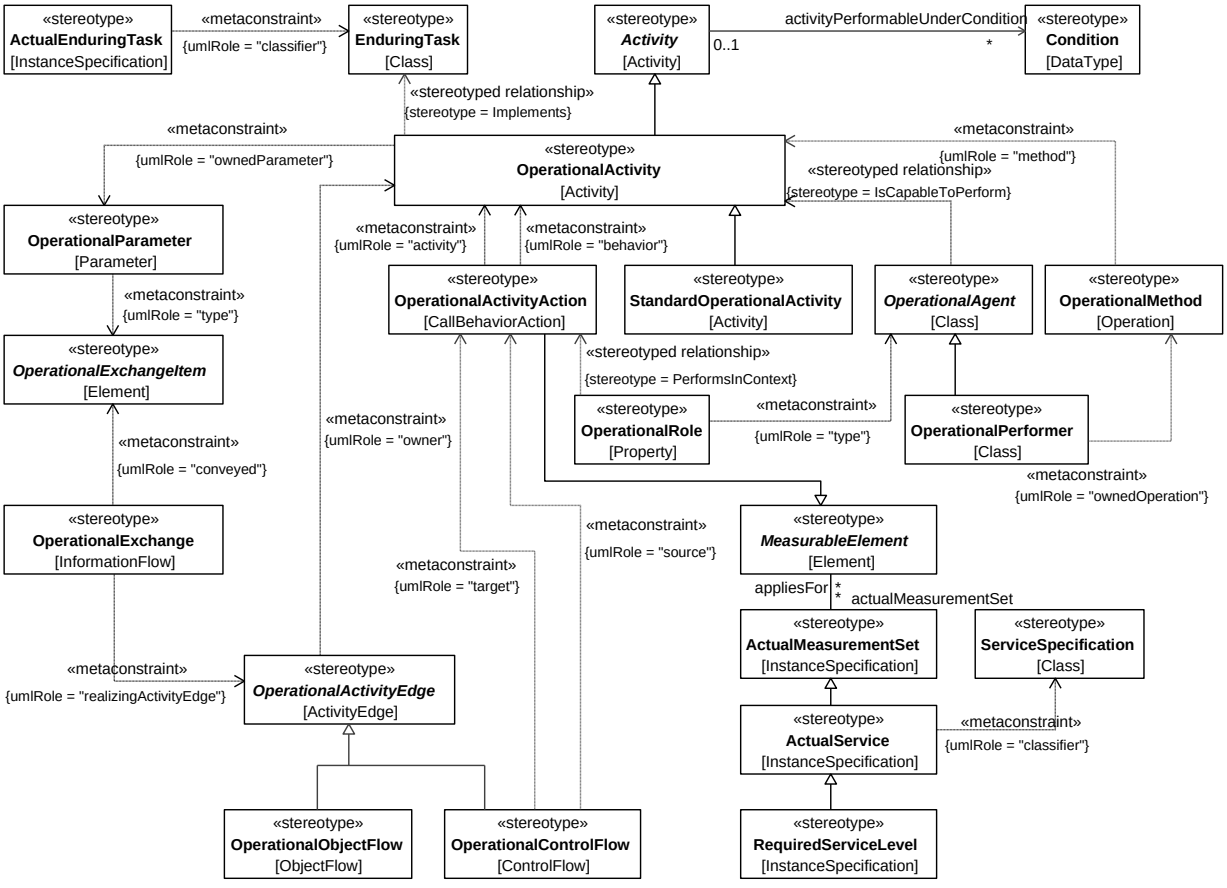

Figure 218 – Operational Processes

Elements

- Activity

- ActualEnduringTask
- ActualMeasurementSet
- ActualService
- Condition
- EnduringTask
- MeasurableElement
- OperationalActivity
- OperationalActivityAction
- OperationalActivityEdge
- OperationalAgent
- OperationalControlFlow
- OperationalExchange
- OperationalExchangeItem
- OperationalMethod
- OperationalObjectFlow
- OperationalParameter
- OperationalPerformer
- OperationalRole
- RequiredServiceLevel
- ServiceSpecification
- StandardOperationalActivity

View Specifications::Operational::States

Stakeholders: Systems Engineers, Software Engineers.
Concerns: capture state-based behavior of an operational OperationalPerformer.
Definition: it is a graphical representation of states of an operational OperationalPerformer and how that operational OperationalPerformer responds to various events and actions.
Recommended Implementation: SysML State Diagram.

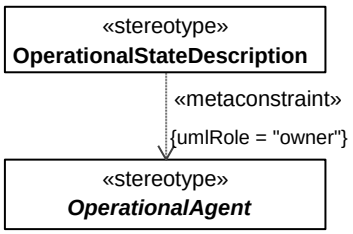

Figure 219 – Operational States

Elements

- OperationalAgent
- OperationalStateDescription

View Specifications::Operational::Interaction Scenarios

Stakeholders: Systems Engineers, Business Architects.
Concerns: express a time ordered examination of the operational exchanges as a result of a particular operational scenario.
Definition: provides a time-ordered examination of the operational exchanges between participating nodes (OperationalPerformer roles) as a result of a particular operational scenario.

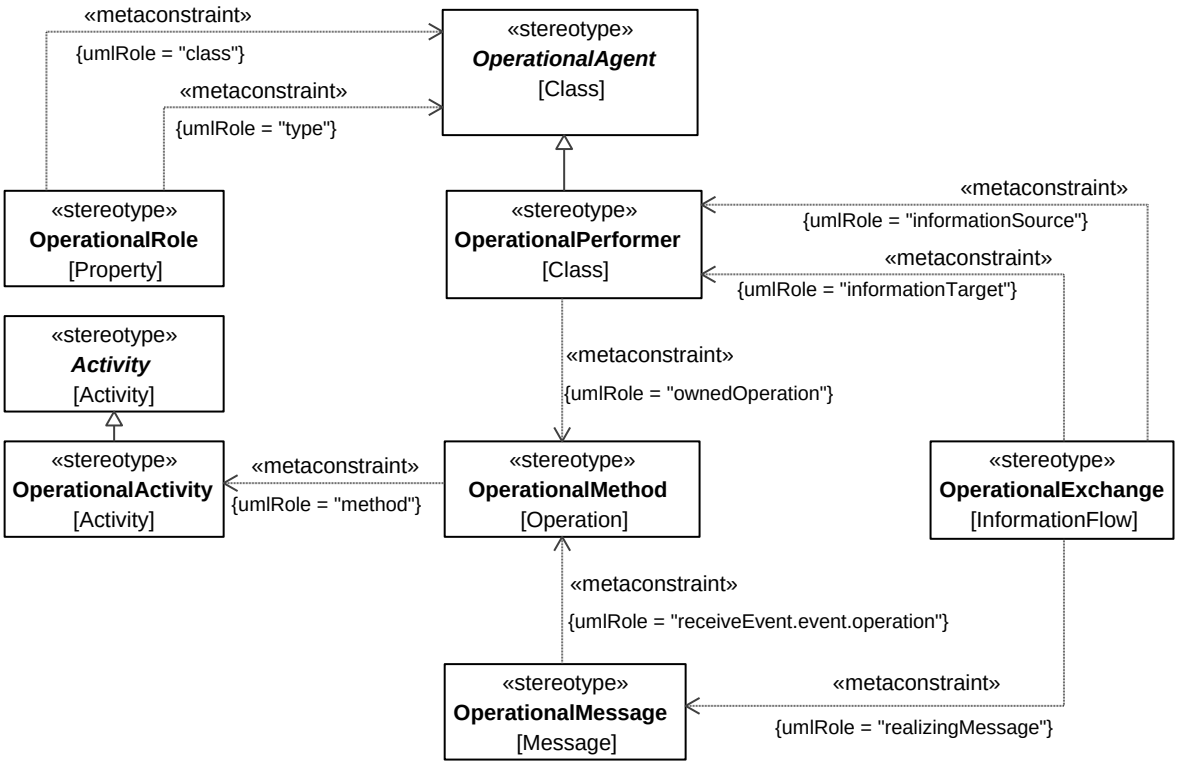

Figure 220 – Operational Interaction Scenarios

Elements

- Activity
- OperationalActivity
- OperationalAgent
- OperationalExchange
- OperationalMessage
- OperationalMethod
- OperationalPerformer
- OperationalRole

View Specifications::Operational::Constraints

Stakeholders: Systems Engineers, Architects, Program Sponsors
Concerns: define operational limitations, constraints and performance parameters for the enterprise.
Definition: specifies traditional textual operational or business rules that are constraints on the way that business is done in the enterprise. The addition of SysML parametrics provides a computational means of defining operational constraints across the enterprise or within a specific operational context.
Recommended Implementation: tabular format, SysML Block Definition Diagram, SysML Parametric Diagram.

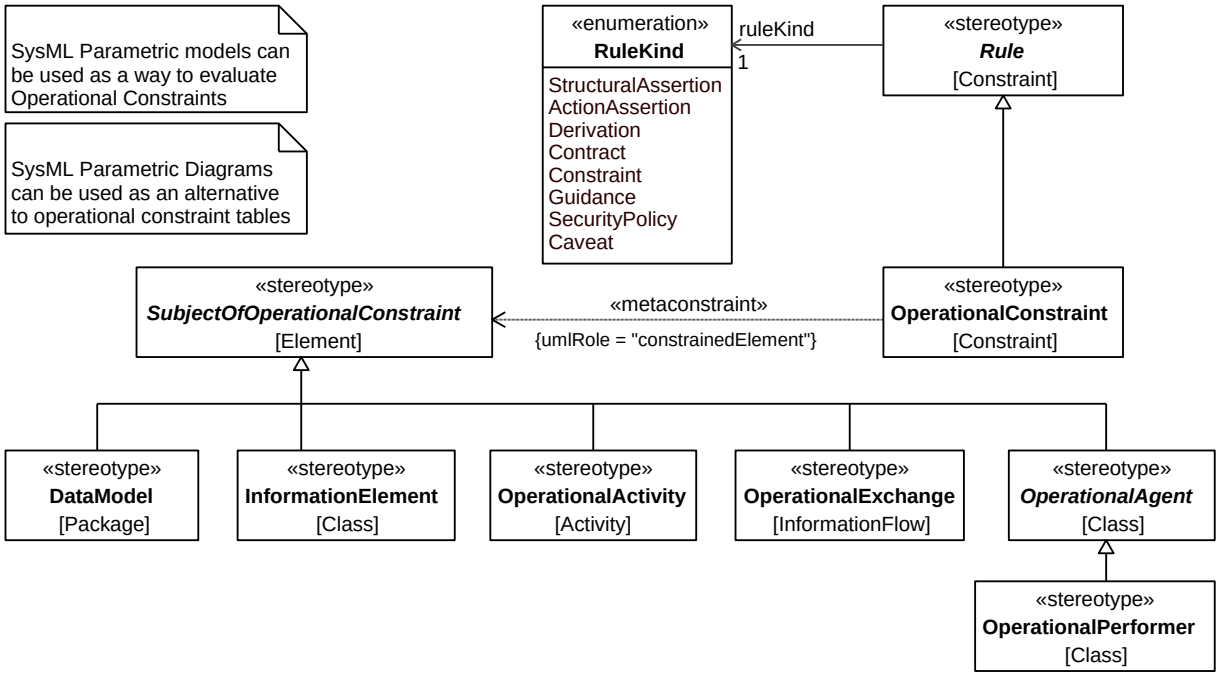

Figure 221 – Operational Constraints

Elements

- DataModel
- InformationElement
- OperationalActivity
- OperationalAgent
- OperationalConstraint
- OperationalExchange
- OperationalPerformer
- Rule
- RuleKind
- SubjectOfOperationalConstraint

View Specifications::Operational::Traceability

Stakeholders: PMs, Enterprise Architects, Business Architects.
Concerns: traceability between capabilities and operational activities and capabilities and operational agents.
Definition: describes the mapping between the capabilities required by an Enterprise and the supporting operational activities and operational agents.
Recommended Implementation: matrix format, SysML Block Definition Diagram.

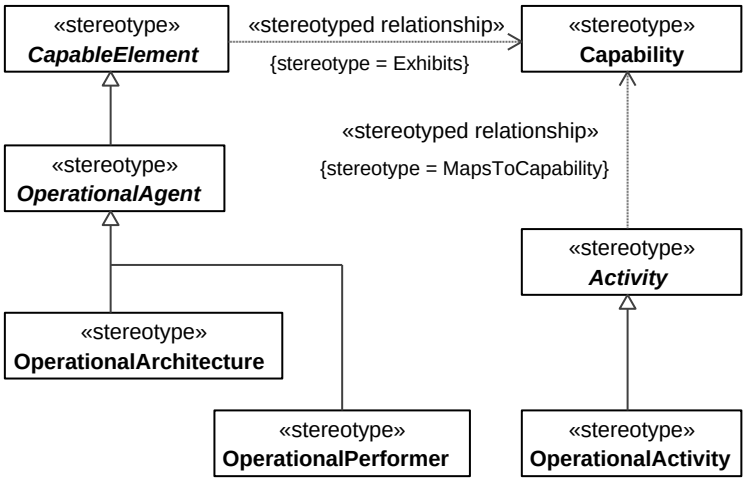

Figure 222 – Operational Traceability

Elements

- Activity
- Capability
- CapableElement
- OperationalActivity
- OperationalAgent
- OperationalArchitecture
- OperationalPerformer

View Specifications::Services

Stakeholders: Enterprise Architects, Solution Providers, Systems Engineers, Software Architects, Business Architects..
Concerns: specifications of services required to exhibit a Capability.
Definition: shows Service Specifications and required and provided service levels of these specifications required to exhibit a Capability or to support an Operational Activity.

View Specifications::Services::Taxonomy

Stakeholders: Enterprise Architects, Solution Providers, Systems Engineers, Software Architects, Business Architects.
Concerns: service specification types and required and provided service levels of these types.
Definition: shows the taxonomy of types of services and the level of service that they are expected to provide or are required to meet through the display of ActualMeasurements associated with the Provided and Required Service Level.
Recommended Implementation: SysML Block Definition Diagram.

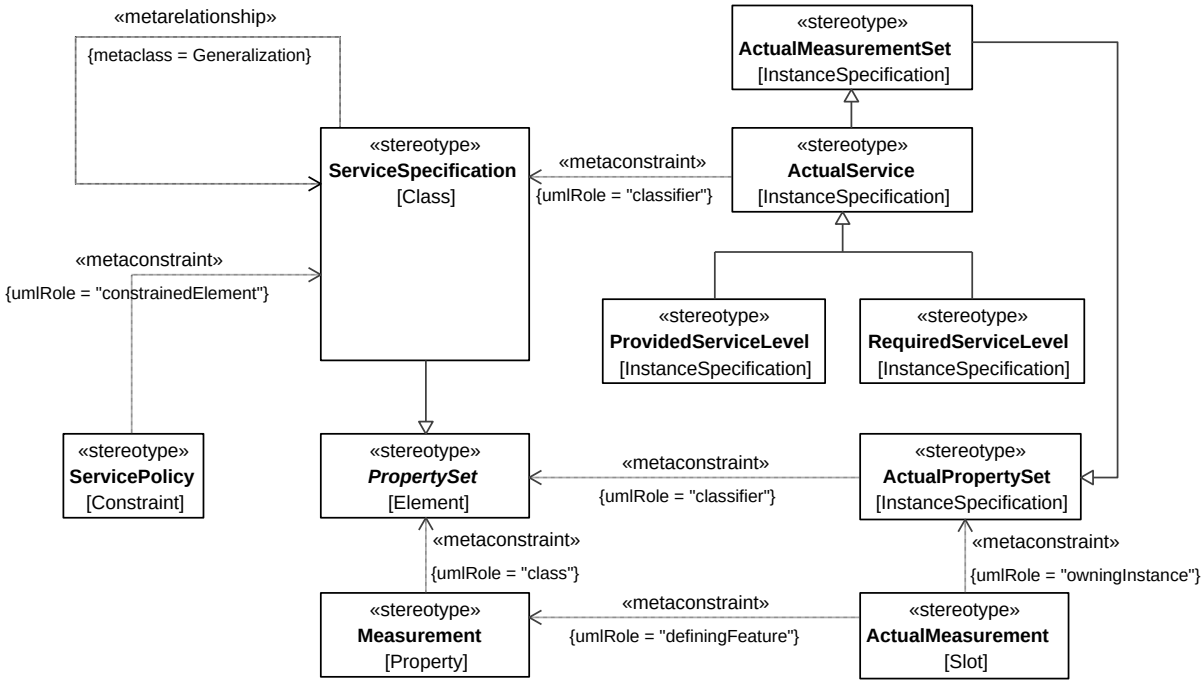

Figure 223 – Services Taxonomy

Elements

- ActualMeasurement
- ActualMeasurementSet
- ActualPropertySet
- ActualService
- Measurement
- PropertySet
- ProvidedServiceLevel
- RequiredServiceLevel
- ServicePolicy
- ServiceSpecification

View Specifications::Services::Structure

Stakeholders: Solution Providers, Systems Engineers, Software Architects, Business Architects.
Concerns: combination of services required to exhibit a capability.
Definition: shows the composition of services and how services are combined into a higher level service required to exhibit a capability or support an operational activity.
Recommended Implementation: SysML Block Definition Diagram, SysML Internal Block Diagram.

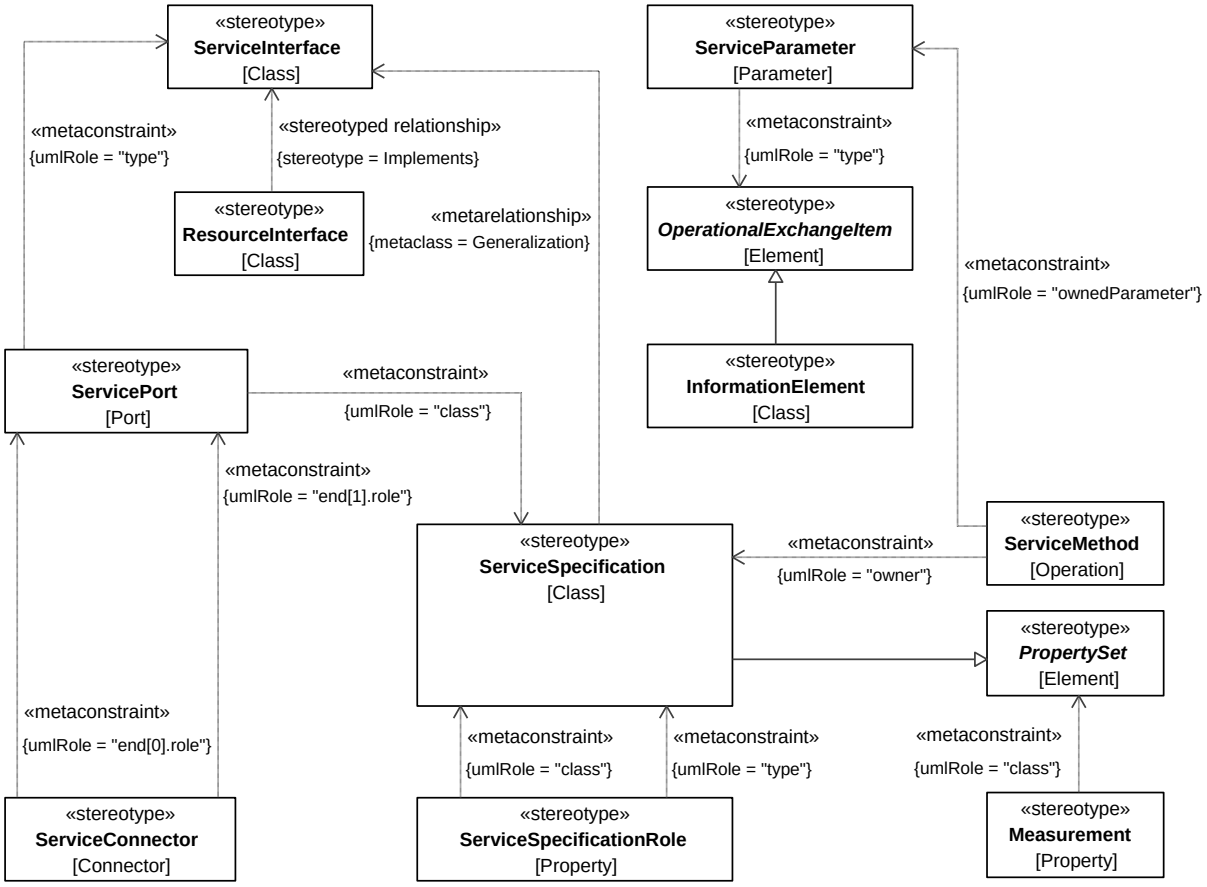

Figure 224 – Services Structure

Elements

- InformationElement
- Measurement
- OperationalExchangeItem
- PropertySet
- ResourceInterface
- ServiceConnector
- ServiceInterface
- ServiceMethod
- ServiceParameter
- ServicePort
- ServiceSpecification
- ServiceSpecificationRole

View Specifications::Services::Connectivity

Stakeholders: Solution Providers, Systems Engineers, Software Architects, Business Architects.
Concerns: interoperability among services
Definition: specifies service interfaces, e.g. provided and required service operations, to ensure compatibility and reusability of services.
Recommended Implementation: SysML Block Definition Diagram, SysML Internal Block Diagram, tabular format.

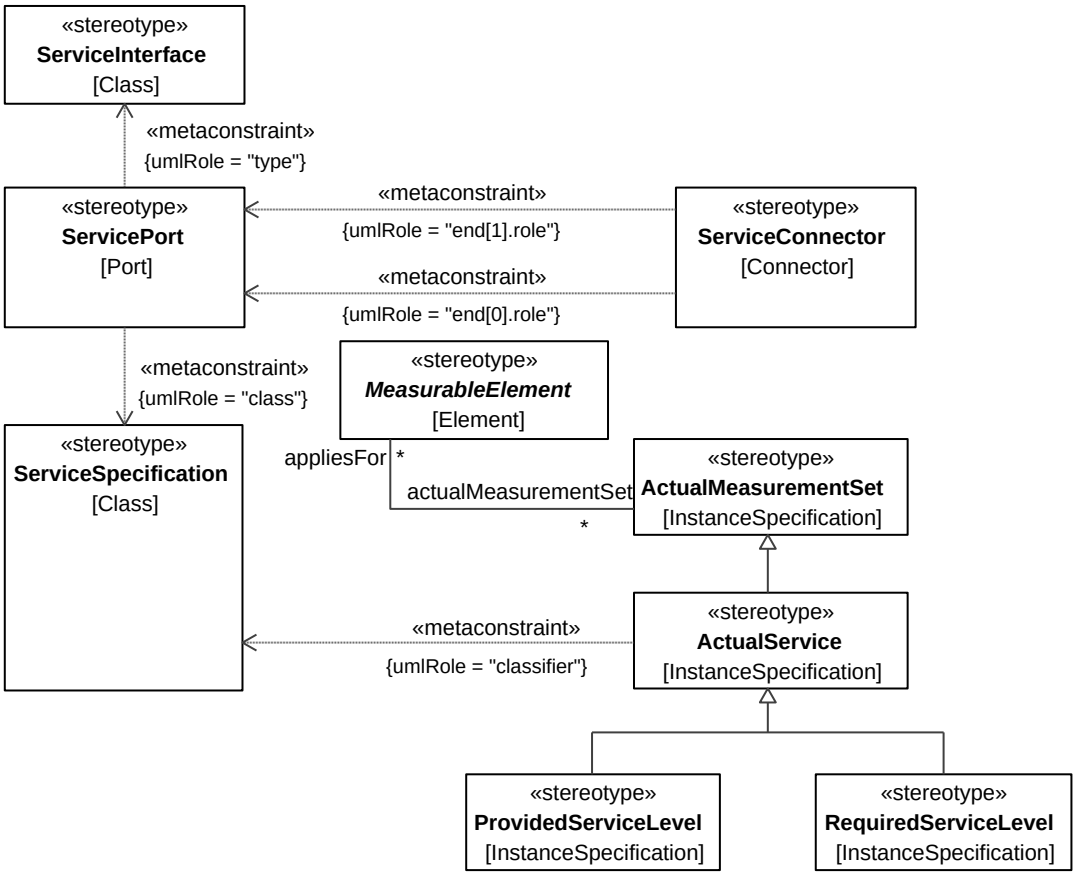

Figure 225 – Services Connectivity

Elements

- ActualMeasurementSet
- ActualService
- MeasurableElement
- ProvidedServiceLevel
- RequiredServiceLevel
- ServiceConnector
- ServiceInterface
- ServicePort
- ServiceSpecification

View Specifications::Services::Processes

Stakeholders: Solution Providers, Systems Engineers, Software Architects, Business Architects.
Concerns: the behavior of a service in terms of the operational activities it is expected to support.
Definition: provides detailed information regarding the allocation of service functions to service specifications, and data flows between service functions.
Recommended Implementation: SysML Block Definition Diagram, SysML Internal Block Diagram, tabular format.

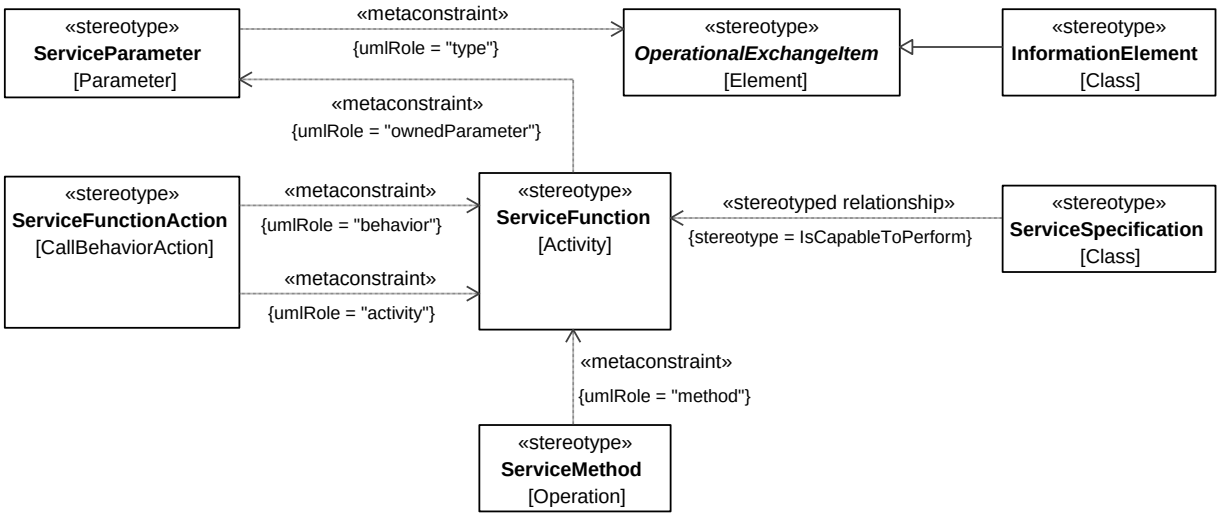

Figure 226 – Services Processes

Elements

- InformationElement
- OperationalExchangeItem
- ServiceFunction
- ServiceFunctionAction
- ServiceMethod
- ServiceParameter
- ServiceSpecification

View Specifications::Services::States

Stakeholders: Solution Providers, Systems Engineers, Software Architects, Business Architects.
Concerns: the behavior of a service specification in terms of states and events causing transitions between states.
Definition: specifies the possible states a service specification may have, and the possible transitions between those states.
Recommended Implementation: SysML State Machine Diagram.

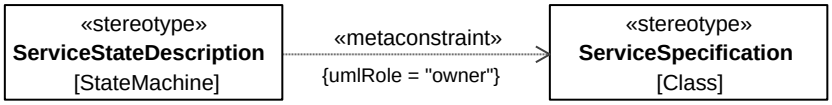

Figure 227 – Services States

Elements

- ServiceSpecification
- ServiceStateDescription

View Specifications::Services::Interaction Scenarios

Stakeholders: Solution Providers, Systems Engineers, Software Architects, Business Architects.
Concerns: the behavior of a service specification in terms of expected time-ordered examination of the interactions between service roles.
Definition: specifies how a service roles interact with each other, service providers and consumers, and the sequence and dependencies of those interactions.
Recommended Implementation: SysML Sequence Diagram.

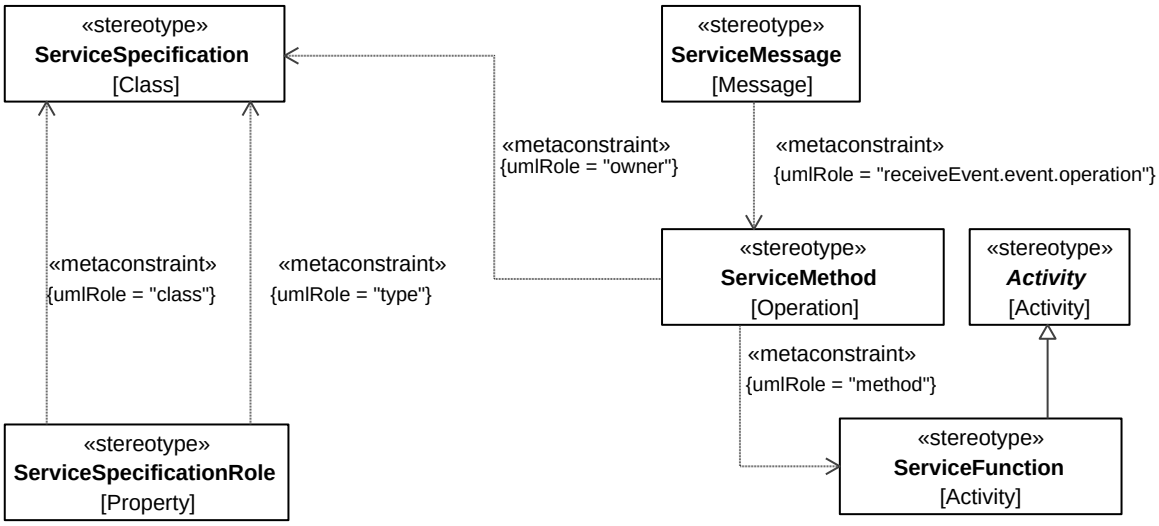

Figure 228 – Services Interaction Scenarios

Elements

- Activity
- ServiceFunction
- ServiceMessage
- ServiceMethod
- ServiceSpecification
- ServiceSpecificationRole

View Specifications::Services::Constraints

Stakeholders: Solution Providers, Systems Engineers, Software Architects, Business Architects.
Concerns: service policies that apply to implementations of service specifications.
Definition: specifies traditional textual service policies that are constraints on the way that service specifications are implemented within resources. The addition of SysML parametrics provide a computational means of defining service policies across the enterprise or within a specific service configuration.
Recommended Implementation: tabular format, SysML Parametric Diagram.

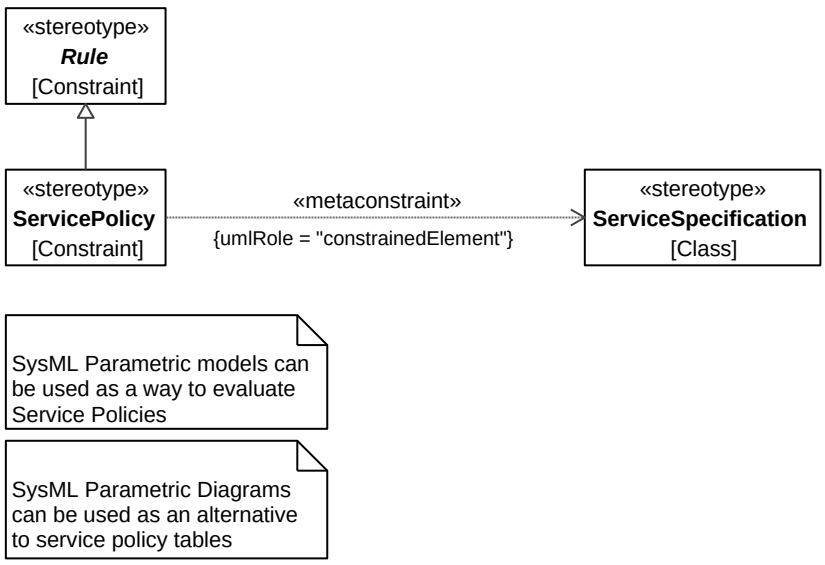

Figure 229 – Services Constraints

Elements

- Rule
- ServicePolicy
- ServiceSpecification

View Specifications::Services::Roadmap

Stakeholders: Solution Providers, Systems Engineers, Software Architects, Business Architects.
Concerns: service specification changes over time.
Definition: provides an overview of how a service specification changes over time. It shows the combination of several service specifications mapped against a timeline.
Recommended Implementation: timeline, SysML Block Definition Diagram, SysML Internal Block Diagram.

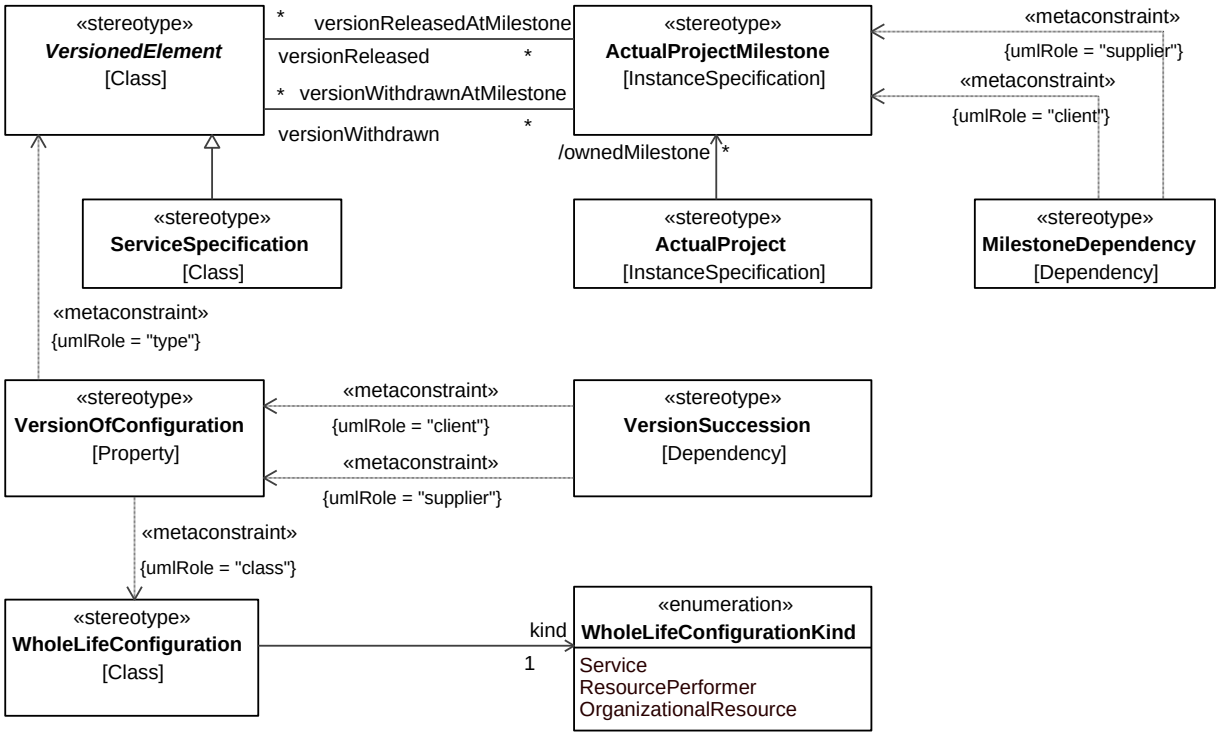

Figure 230 – Services Roadmap

Elements

- ActualProject
- ActualProjectMilestone
- MilestoneDependency
- ServiceSpecification
- VersionedElement
- VersionOfConfiguration
- VersionSuccession
- WholeLifeConfiguration
- WholeLifeConfigurationKind

View Specifications::Services::Traceability

Stakeholders: Solution Providers, Systems Engineers, Software Architects, Business Architects.
Concerns: traceability between operational activities and service specifications that support them.
Definition: depicts the mapping of service specifications to operational activities and how service specifications contribute to the achievement of a capability.
Recommended Implementation: timeline, SysML Block Definition Diagram, SysML Internal Block Diagram.

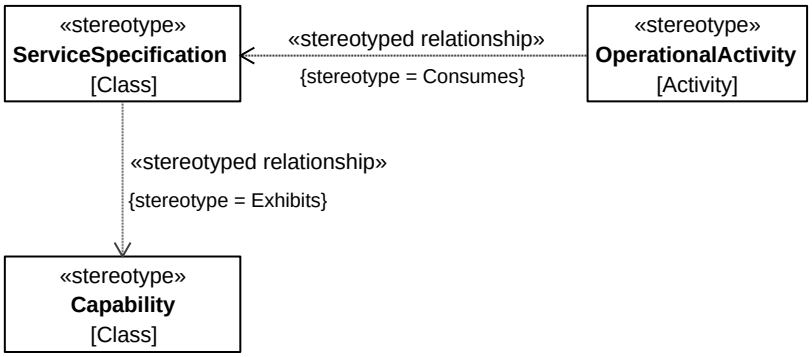

Figure 231 – Services Traceability

Elements

- Capability
- OperationalActivity
- ServiceSpecification

View Specifications::Personnel

Stakeholders: Human resources, Solution Providers, PMs.
Concerns: human factors.
Definition: aims to clarify the role of Human Factors (HF) when creating architectures in order to facilitate both Human Factors Integration (HFI) and systems engineering (SE).

View Specifications::Personnel::Taxonomy

Stakeholders: Human resources, Solution Providers, PMs.
Concerns: organizational resource types.
Definition: shows the taxonomy of types of organizational resources.
Recommended Implementation: SysML Block Definition Diagram.

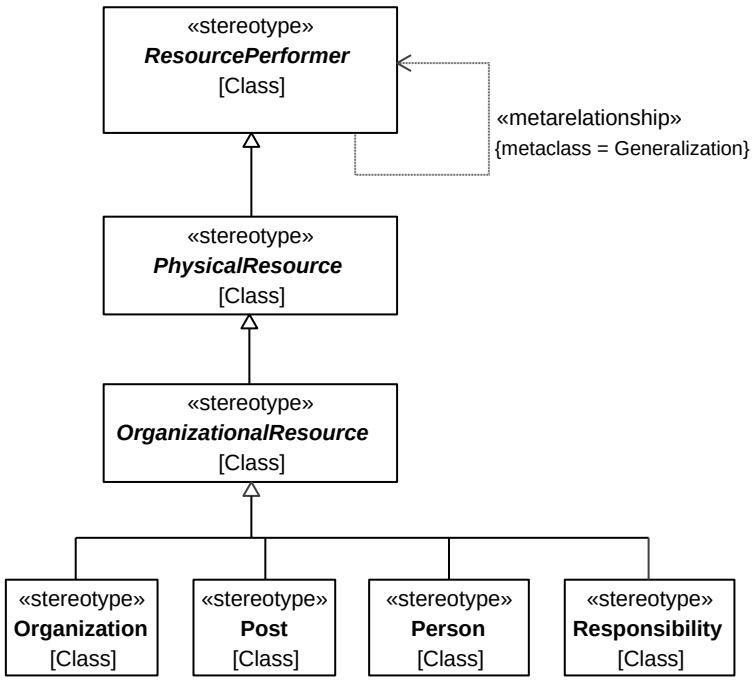

Figure 232 – Personnel Taxonomy

Elements

- Organization

- OrganizationalResource
- Person
- PhysicalResource
- Post
- ResourcePerformer
- Responsibility

View Specifications::Personnel::Structure

Stakeholders: Human resources, Solution Providers, PMs.
Concerns: typical organizational structure used to support a capability(ies).
Definition: shows organizational structures and possible interactions between organizational resources.
Recommended Implementation: SysML Block Definition Diagram, SysML Internal Block Diagram.

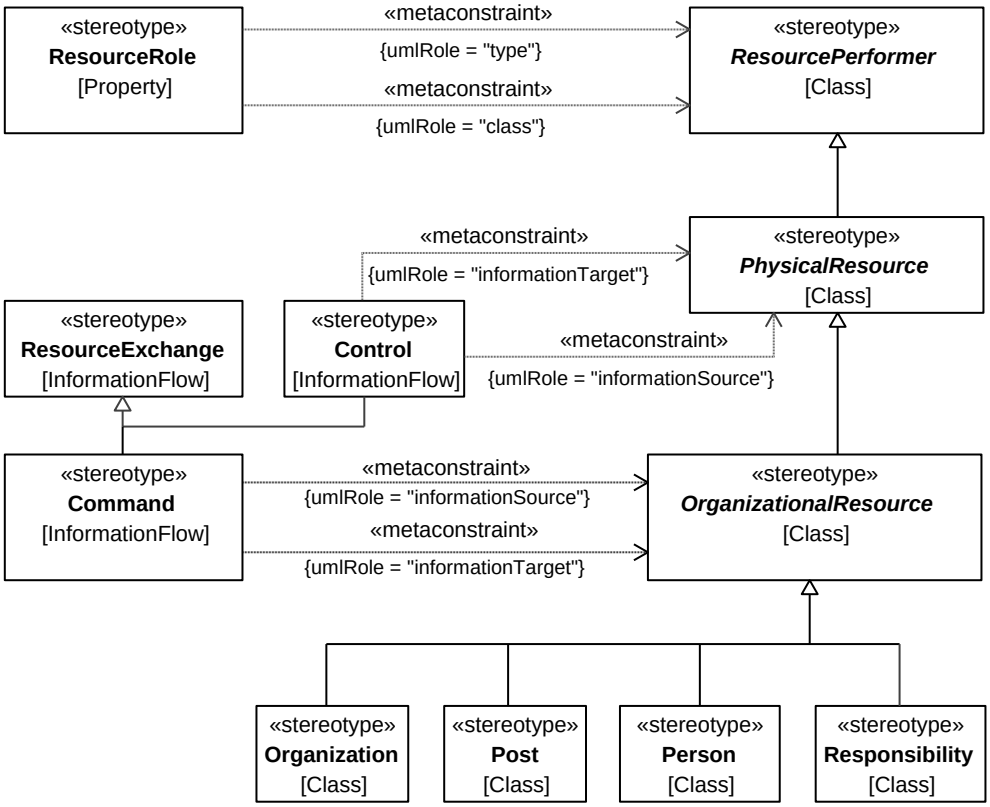

Figure 233 – Personnel Sructure

Elements

- Command
- Control
- Organization
- OrganizationalResource
- Person
- PhysicalResource
- Post
- ResourceExchange
- ResourcePerformer
- ResourceRole
- Responsibility

View Specifications::Personnel::Connectivity

Stakeholders: Solution providers.
Concerns: interaction of organizational resources.
Definition: captures the possible interactions between organizational resources, including command and control relationships. Interactions typically illustrate the fundamental roles and management responsibilities.
Recommended Implementation: tabular format.

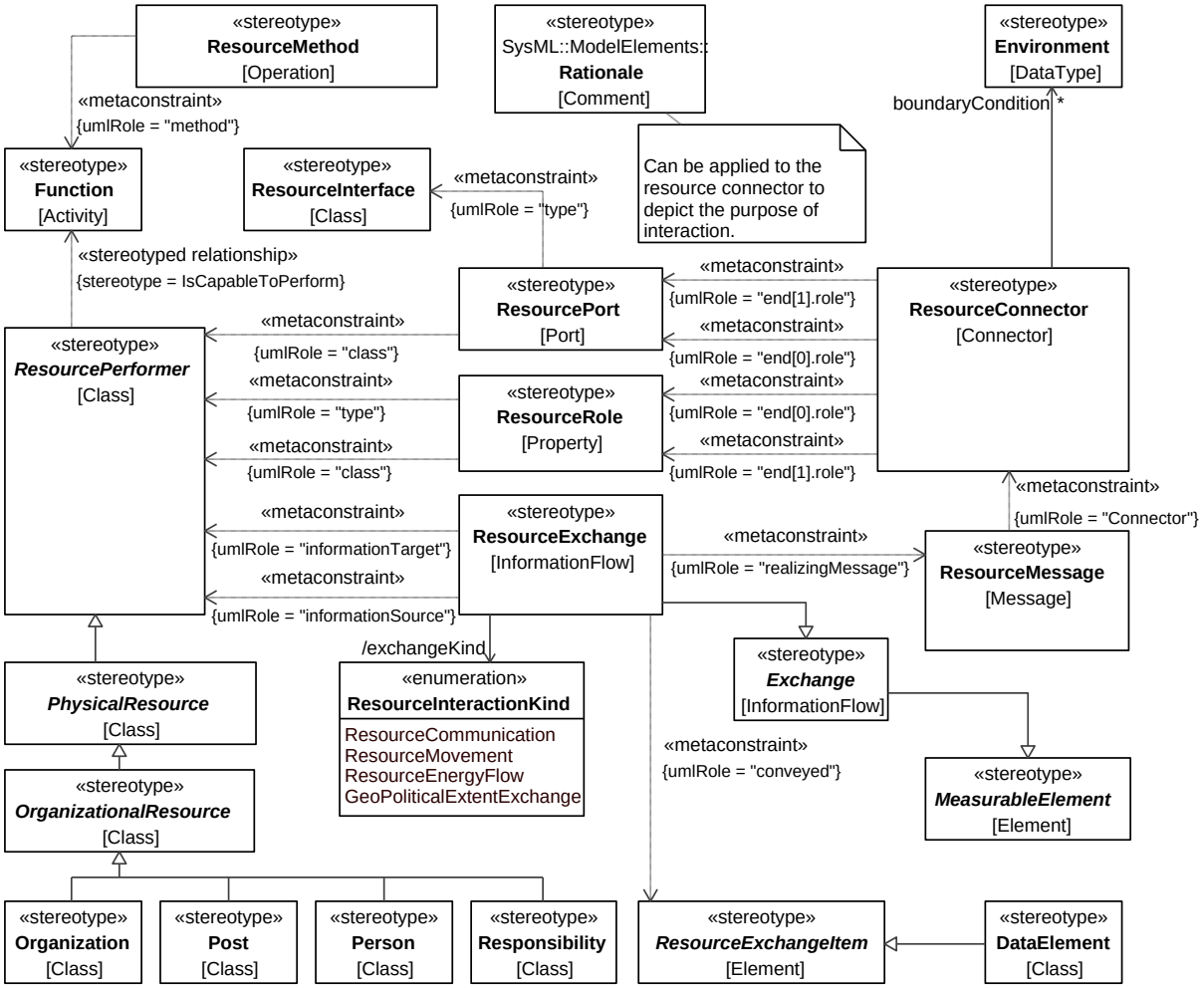

Figure 234 – Personnel Connectivity

Elements

- DataElement
- Environment
- Exchange
- Function
- MeasurableElement
- Organization
- OrganizationalResource
- Person
- PhysicalResource
- Post
- Rationale
- ResourceConnector
- ResourceExchange
- ResourceExchangeItem

- ResourceInteractionKind
- ResourceInterface
- ResourceMessage
- ResourceMethod
- ResourcePerformer
- ResourcePort
- ResourceRole
- Responsibility

View Specifications::Personnel::Processes

Stakeholders: Systems engineers, Solution providers.
Concerns: functions that have to be carried out by organizational resources.
Definition: specifies organizational resource functions in relation to resource definitions.
Recommended Implementation: SysML Activity Diagram, SysML Block Definition Diagram, BPMN Process Diagram.

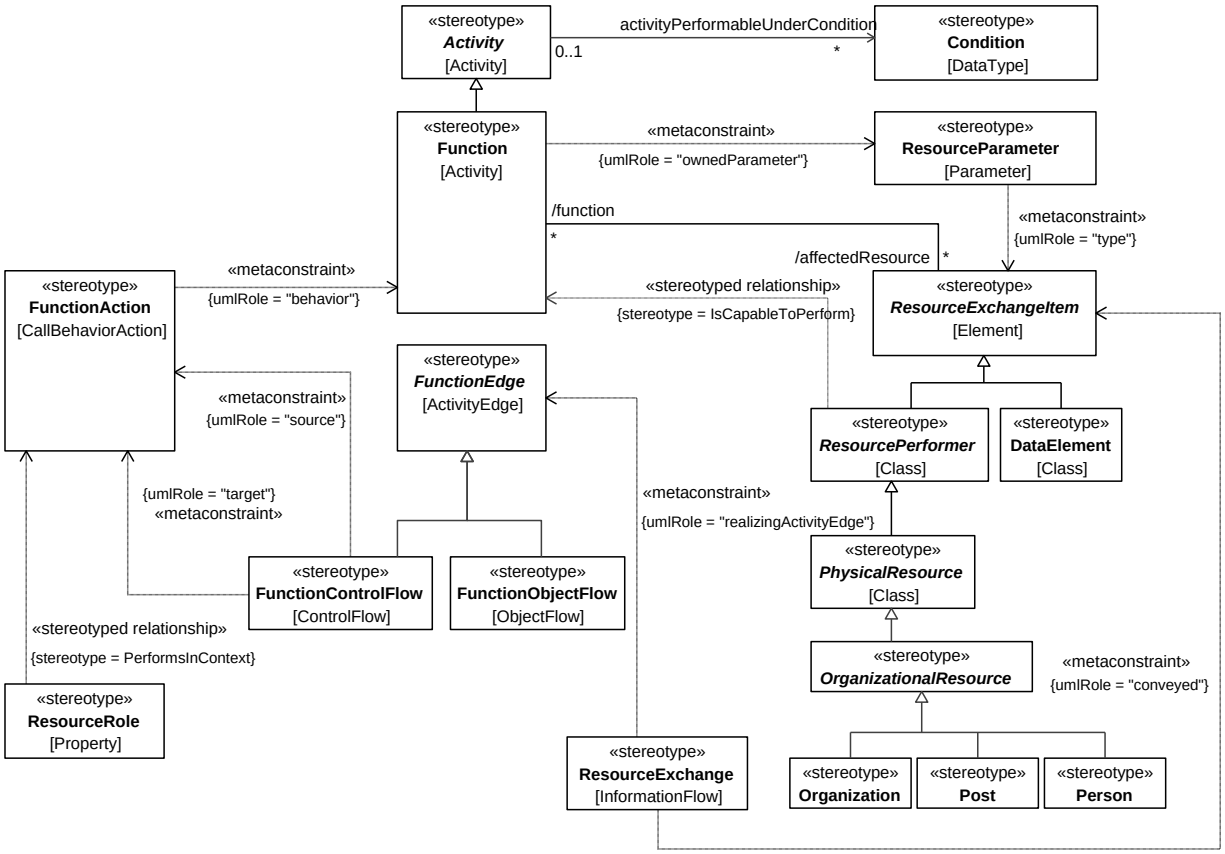

Figure 235 – Personnel Processes

Elements

- Activity
- Condition
- DataElement
- Function
- FunctionAction
- FunctionControlFlow
- FunctionEdge
- FunctionObjectFlow
- Organization
- OrganizationalResource

- Person
- PhysicalResource
- Post
- ResourceExchange
- ResourceExchangeItem
- ResourceParameter
- ResourcePerformer
- ResourceRole

View Specifications::Personnel::States

Stakeholders: Systems Engineers, Software Engineers.
Concerns: capture state-based behavior of an organizational resource.
Definition: it is a graphical representation of states of an organizational resource and how that organizational resource responds to various events and actions.
Recommended Implementation: SysML State Diagram.

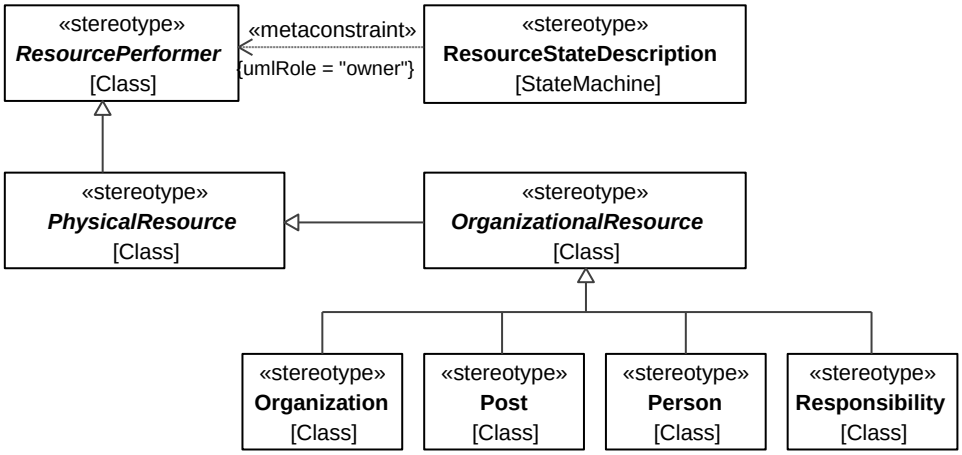

Figure 236 – Personnel States

Elements

- Organization
- OrganizationalResource
- Person
- PhysicalResource
- Post
- ResourcePerformer
- ResourceStateDescription
- Responsibility

View Specifications::Personnel::Interaction Scenarios

Stakeholders: Software Engineers, Systems Engineers.
Concerns: interactions between organizational resources (roles).
Definition: provides a time-ordered examination of the interactions between organizational resources.
Recommended Implementation: SysML Sequence Diagram, BPMN Collaboration Diagram.

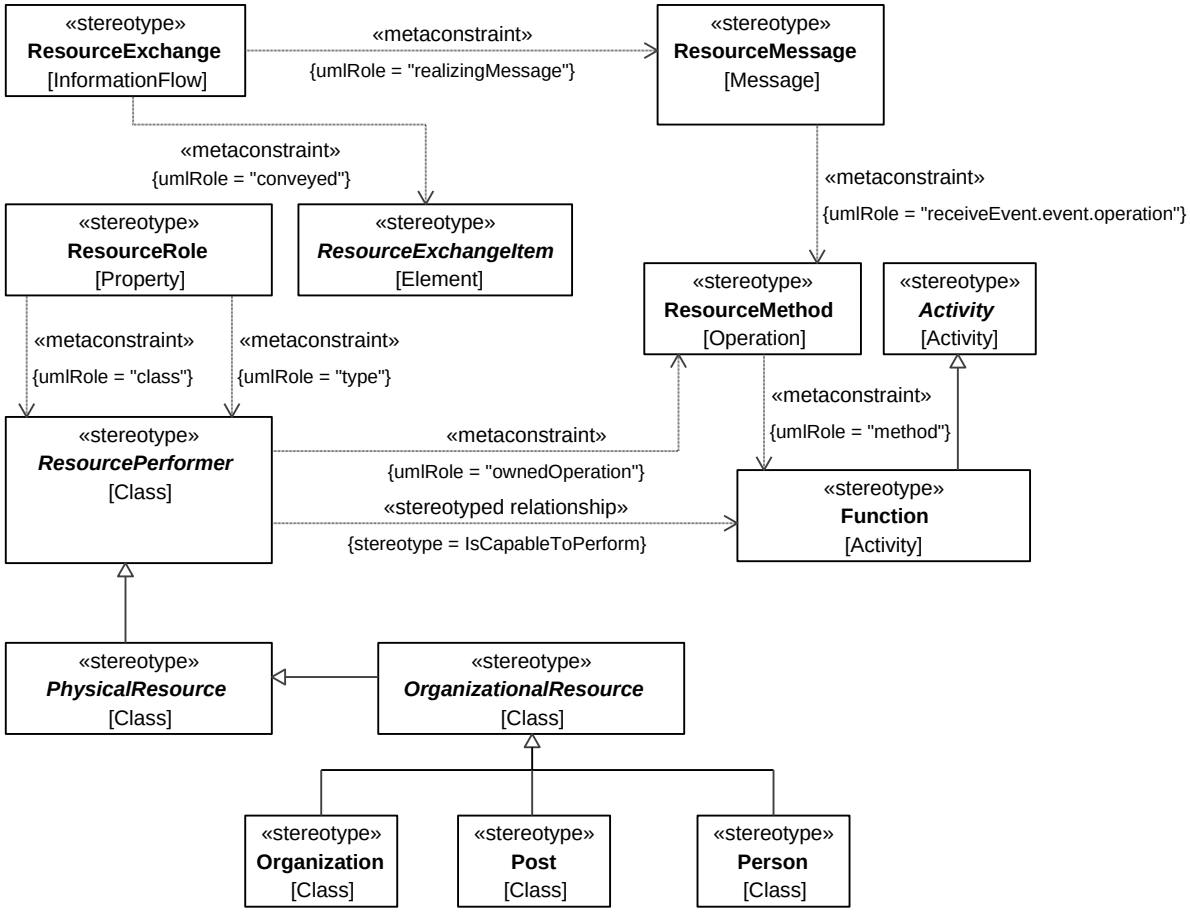

Figure 237 – Personnel Interaction Scenarios

Elements

- Activity
- Function
- Organization
- OrganizationalResource
- Person
- PhysicalResource
- Post
- ResourceExchange
- ResourceExchangeItem
- ResourceMessage
- ResourceMethod
- ResourcePerformer
- ResourceRole

View Specifications::Personnel::Constraints::Personnel Constraints: Competence

Stakeholders: Systems engineers, Solution providers.
Concerns: allocation of competencies to actual posts.
Definition: specifies requirements for actual organizational resources – by linking competencies and actual posts.
Recommended Implementation: SysML Block Definition Diagram.

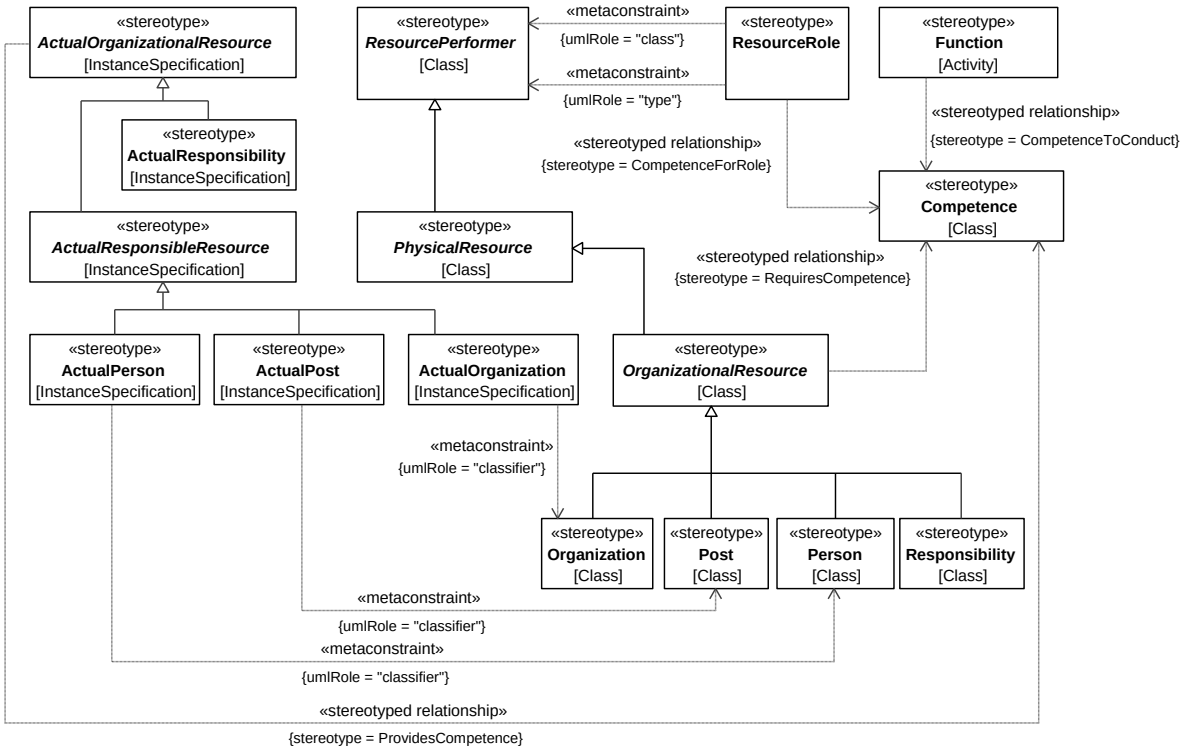

Figure 238 – Personnel Constraints: Competence

Elements

- ActualOrganization
- ActualOrganizationalResource
- ActualPerson
- ActualPost
- ActualResponsibility
- ActualResponsibleResource
- Competence
- Function
- Organization
- OrganizationalResource
- Person
- PhysicalResource
- Post
- ResourcePerformer
- ResourceRole
- Responsibility

View Specifications::Personnel::Constraints::Personnel Constraints: Drivers

Stakeholders: Systems engineers, Solution providers, Human resources.
Concerns: optimization of organizational resource behavior.
Definition: captures the factors that affect, constrain and characterize organizational resource behavior as the basis for performance predictions at the level of actual persons and actual organizations. It creates a bridge between static architectural definitions and behavior predictions through executable models.
Recommended Implementation: tabular format, SysML Parametric Diagram, SysML Block Definition Diagram.

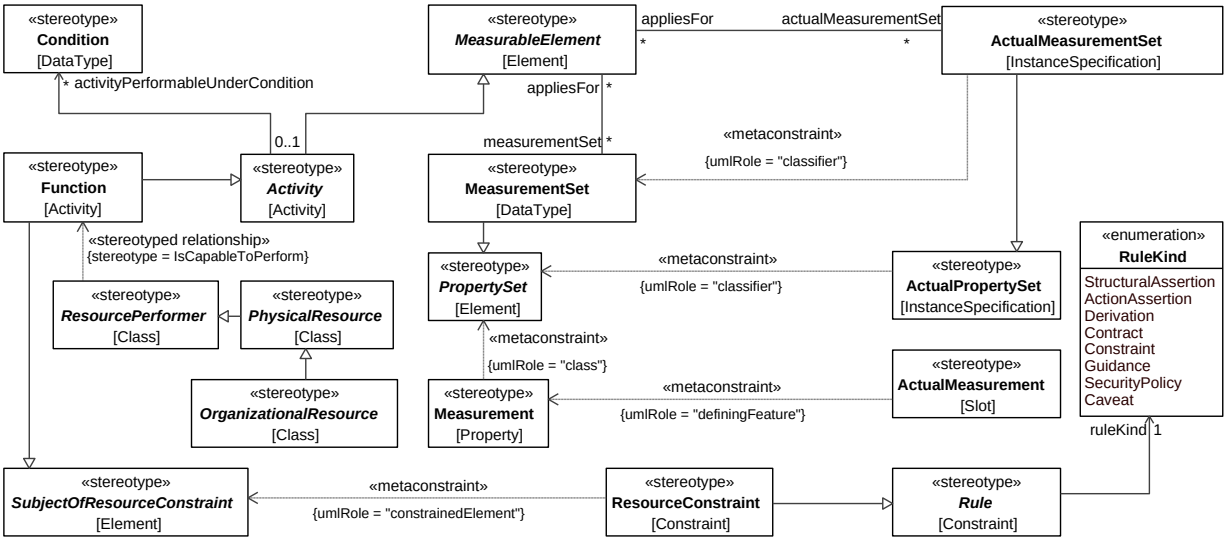

Figure 239 – Personnel Constraints: Drivers

Elements

- Activity
- ActualMeasurement
- ActualMeasurementSet
- ActualPropertySet
- Condition
- Function
- MeasurableElement
- Measurement
- MeasurementSet
- OrganizationalResource
- PhysicalResource
- PropertySet
- ResourceConstraint
- ResourcePerformer
- Rule
- RuleKind
- SubjectOfResourceConstraint

View Specifications::Personnel::Constraints::Personnel Constraints: Performance

Stakeholders: Human resources, solution providers.
Concerns: how well an actual organizational resource matches the needs of the actual organization.
Definition: provides a repository for human-related measures (i.e. quality objectives and performance criteria (HFI values)), targets and competences.
Recommended Implementation: SysML Block Definition Diagram.

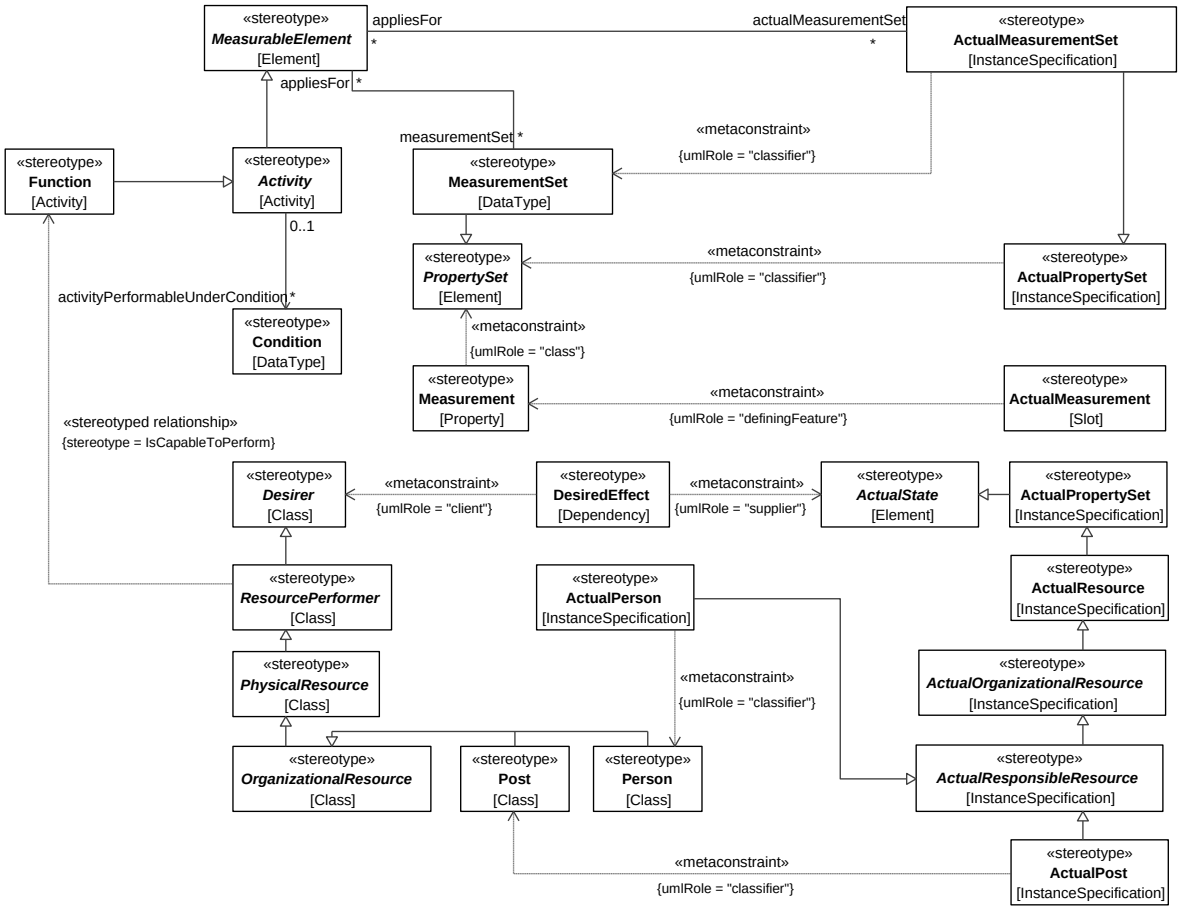

Figure 240 – Personnel Constraints: Performance

Elements

- Activity
- ActualMeasurement
- ActualMeasurementSet
- ActualOrganizationalResource
- ActualPerson
- ActualPost
- ActualPropertySet
- ActualResource
- ActualResponsibleResource
- ActualState
- Condition
- DesiredEffect
- Desirer
- Function
- MeasurableElement
- Measurement
- MeasurementSet
- OrganizationalResource
- Person
- PhysicalResource
- Post
- PropertySet
- ResourcePerformer

View Specifications::Personnel::Roadmap::Personnel Roadmap: Availability

Stakeholders: Human Resources, Training, Logisticians, Solution Providers.
Concerns: the staffing and training of resources.
Definition: defines the requirements and functions to ensure that actual persons with the right competencies, and in the right numbers, are available to fulfill actual posts.
Recommended Implementation: Timeline, SysML Block Definition Diagram.

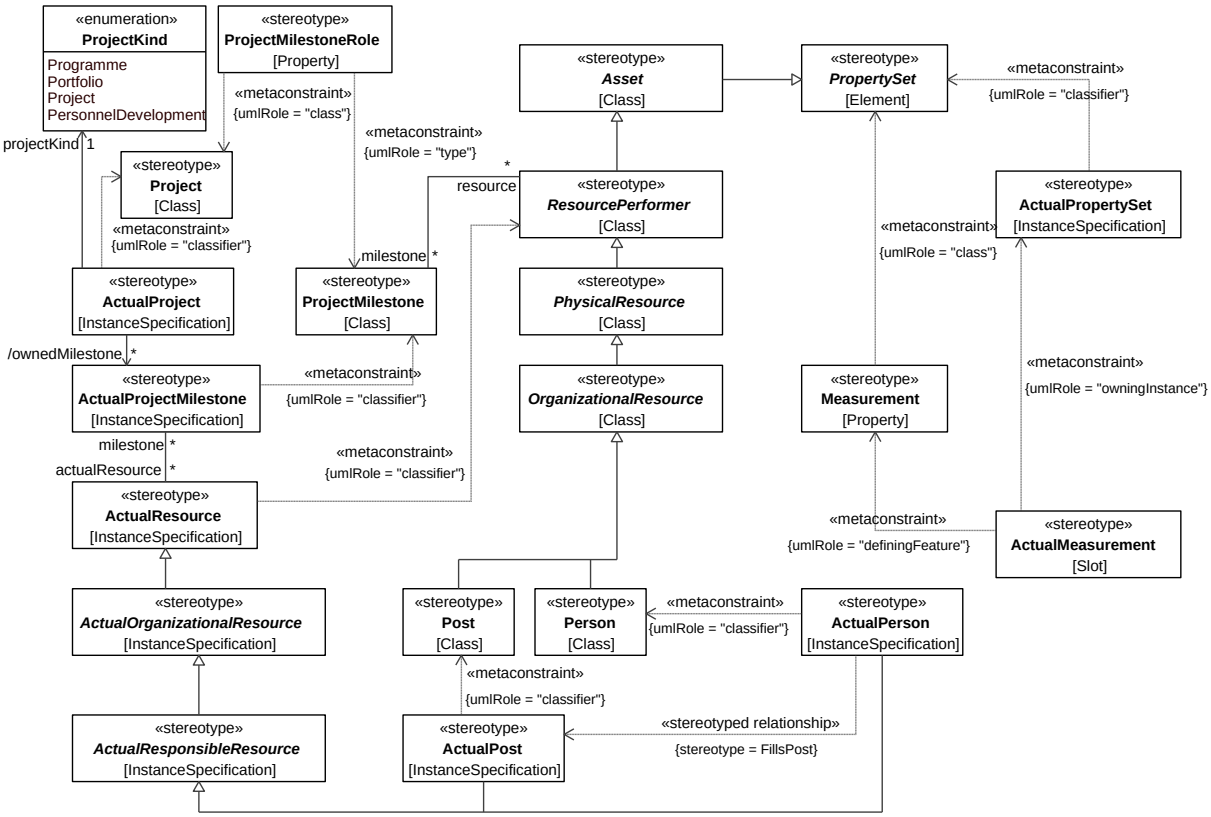

Figure 241 – Personnel Roadmap: Availability

Elements

- ActualMeasurement
- ActualOrganizationalResource
- ActualPerson
- ActualPost
- ActualProject
- ActualProjectMilestone
- ActualPropertySet
- ActualResource
- ActualResponsibleResource
- Asset
- Measurement
- OrganizationalResource
- Person
- PhysicalResource
- Post
- Project
- ProjectKind
- ProjectMilestone
- ProjectMilestoneRole
- PropertySet

- ResourcePerformer

View Specifications::Personnel::Roadmap::Personnel Roadmap: Evolution

Stakeholders: Human resources, Solution Providers.
Concerns: organizational structure changes over time.
Definition: provides an overview of how a organizational structure changes over time. It shows the structure of several organizational structures mapped against a timeline.
Recommended Implementation: timeline, SysML Block Definition Diagram, SysML Internal Block Diagram.

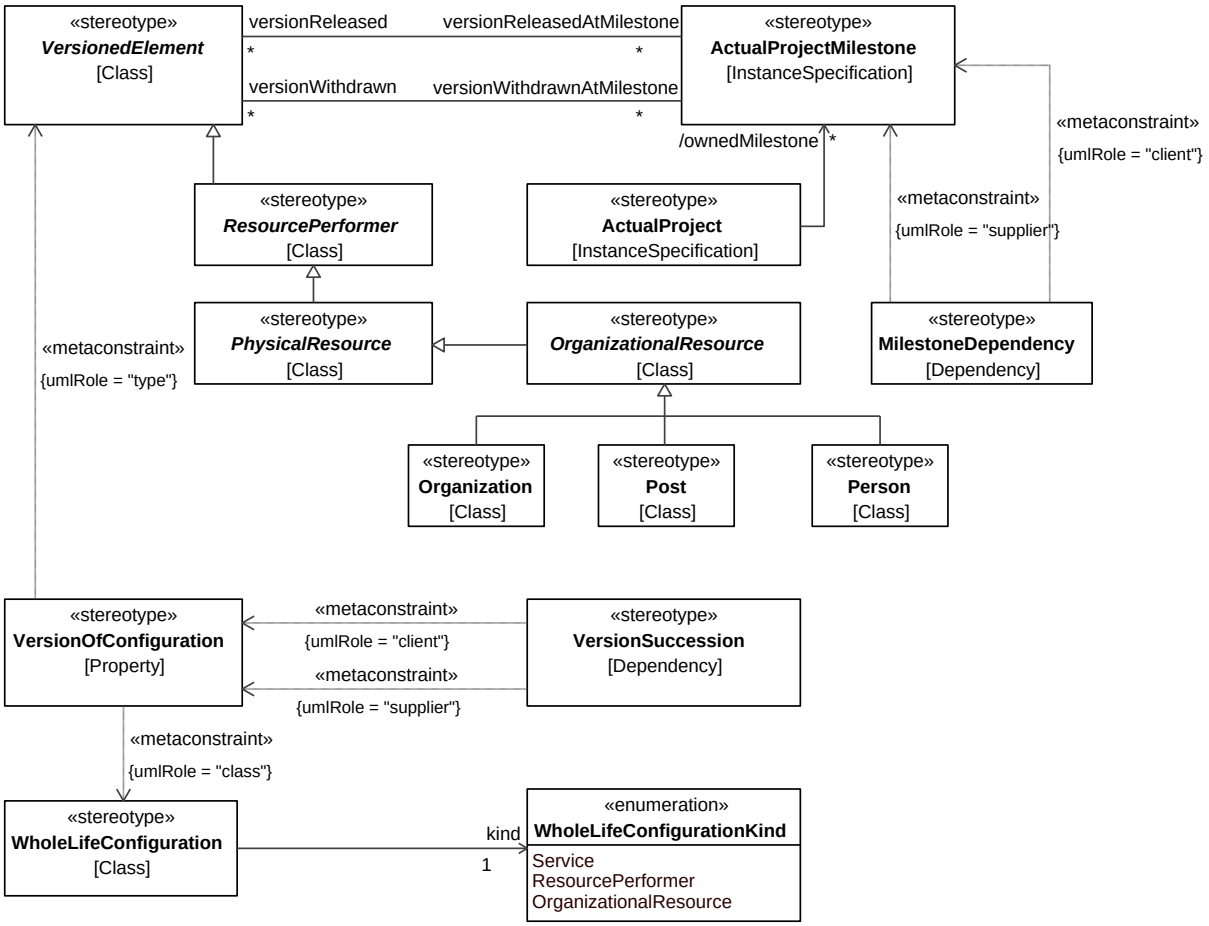

Figure 242 – Personnel Roadmap: Evolution

Elements

- ActualProject
- ActualProjectMilestone
- MilestoneDependency
- Organization
- OrganizationalResource
- Person
- PhysicalResource
- Post
- ResourcePerformer
- VersionedElement
- VersionOfConfiguration
- VersionSuccession
- WholeLifeConfiguration

- WholeLifeConfigurationKind

View Specifications::Personnel::Roadmap::Personnel Roadmap: Forecast

Stakeholders: Human resources, Logisticians, Solution Providers.
Concerns: competencies and skills forecast.
Definition: defines the underlying current and expected supporting competencies and skills of organizational resources.
Recommended Implementation: timeline, tabular format, SysML Block Definition Diagram.

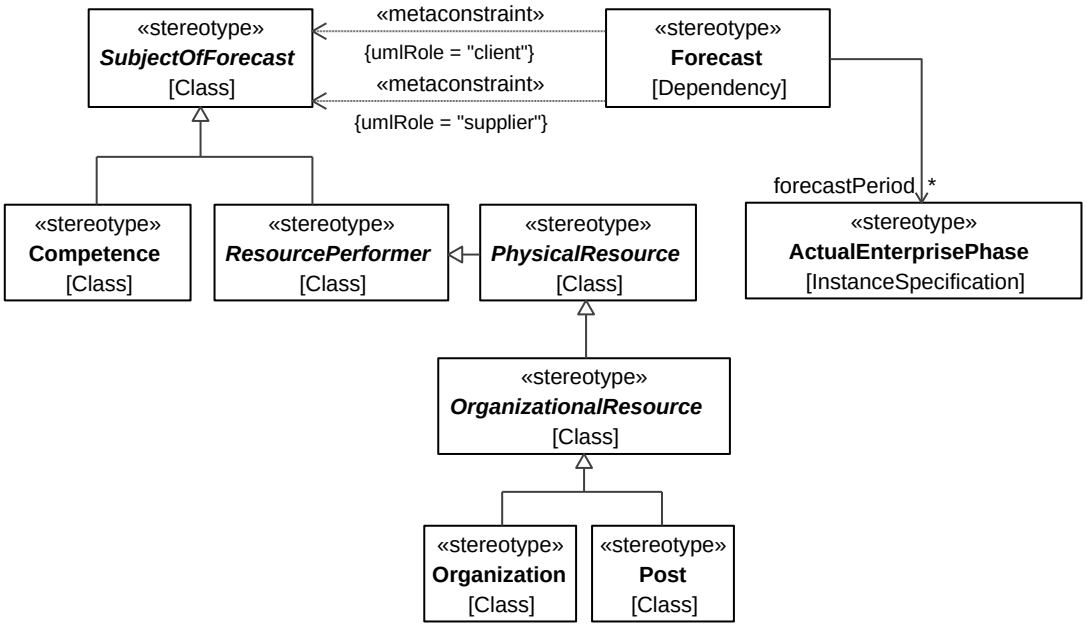

Figure 243 – Personnel Roadmap: Forecast

Elements

- ActualEnterprisePhase
- Competence
- Forecast
- Organization
- OrganizationalResource
- PhysicalResource
- Post
- ResourcePerformer
- SubjectOfForecast

View Specifications::Personnel::Traceability

Stakeholders: Systems Engineers, Enterprise Architects, Solution Providers, Business Architects.
Concerns: traceability between operational activities and functions that implements them.
Definition: depicts the mapping of functions (performed by organizational resources) to operational activities and thus identifies the transformation of an operational need into a purposeful function performed by an organizational resource or solution.
Recommended Implementation: Matrix format, SysML Block Definition Diagram.

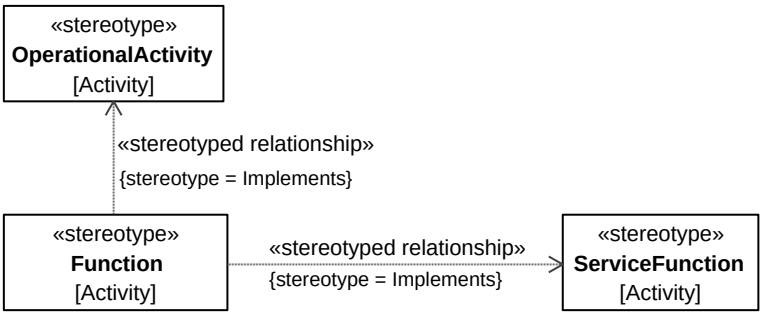

Figure 244 – Personnel Traceability

Elements

- Function
- OperationalActivity
- ServiceFunction

View Specifications::Resources

Stakeholders: Systems Engineers, Resource Owners, Implementers, Solution Providers, IT Architects.
Concerns: definition of solution architectures to implement operational requirements.
Definition: captures a solution architecture consisting of resources, e.g. organizational, software, artifacts, capability configurations, natural resources that implement the operational requirements. Further design of a resource is typically detailed in SysML or UML.

View Specifications::Resources::Taxonomy

Stakeholders: Solution Providers, Systems Engineers, IT Architects, Implementers.
Concerns: resource types.
Definition: shows the taxonomy of types of resources.
Recommended Implementation: SysML Block Definition Diagram.

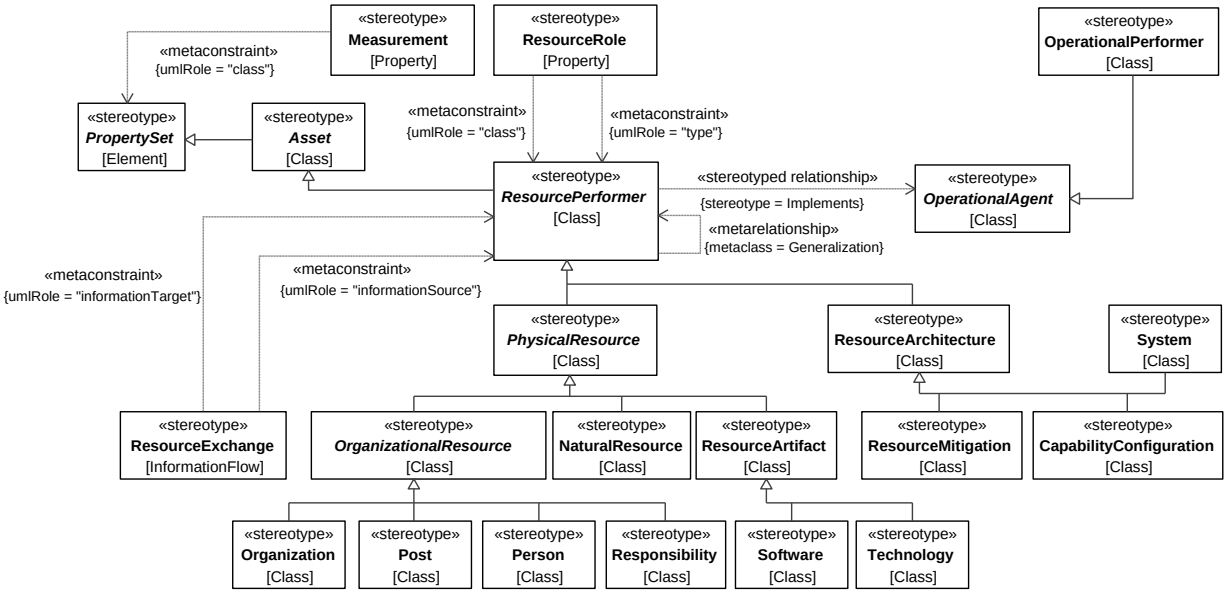

Figure 245 – Resources Taxonomy

Elements

- Asset
- CapabilityConfiguration
- Measurement
- NaturalResource

- OperationalAgent
- OperationalPerformer
- Organization
- OrganizationalResource
- Person
- PhysicalResource
- Post
- PropertySet
- ResourceArchitecture
- ResourceArtifact
- ResourceExchange
- ResourceMitigation
- ResourcePerformer
- ResourceRole
- Responsibility
- Software
- System
- Technology

View Specifications::Resources::Structure

Stakeholders: Systems Engineers, Resource Owners, Implementers, Solution Providers.
Concerns: reference the resource structure, connectors and interfaces in a specific context.
Definition: defines the physical resources, e.g. capability configuration(s)/system(s) and interactions necessary to implement a specific set of OperationalPerformer(s). Can be used to represent communications networks and pathways that link communications resources and provides details regarding their configuration.
Recommended Implementation: SysML Internal Block Diagram.

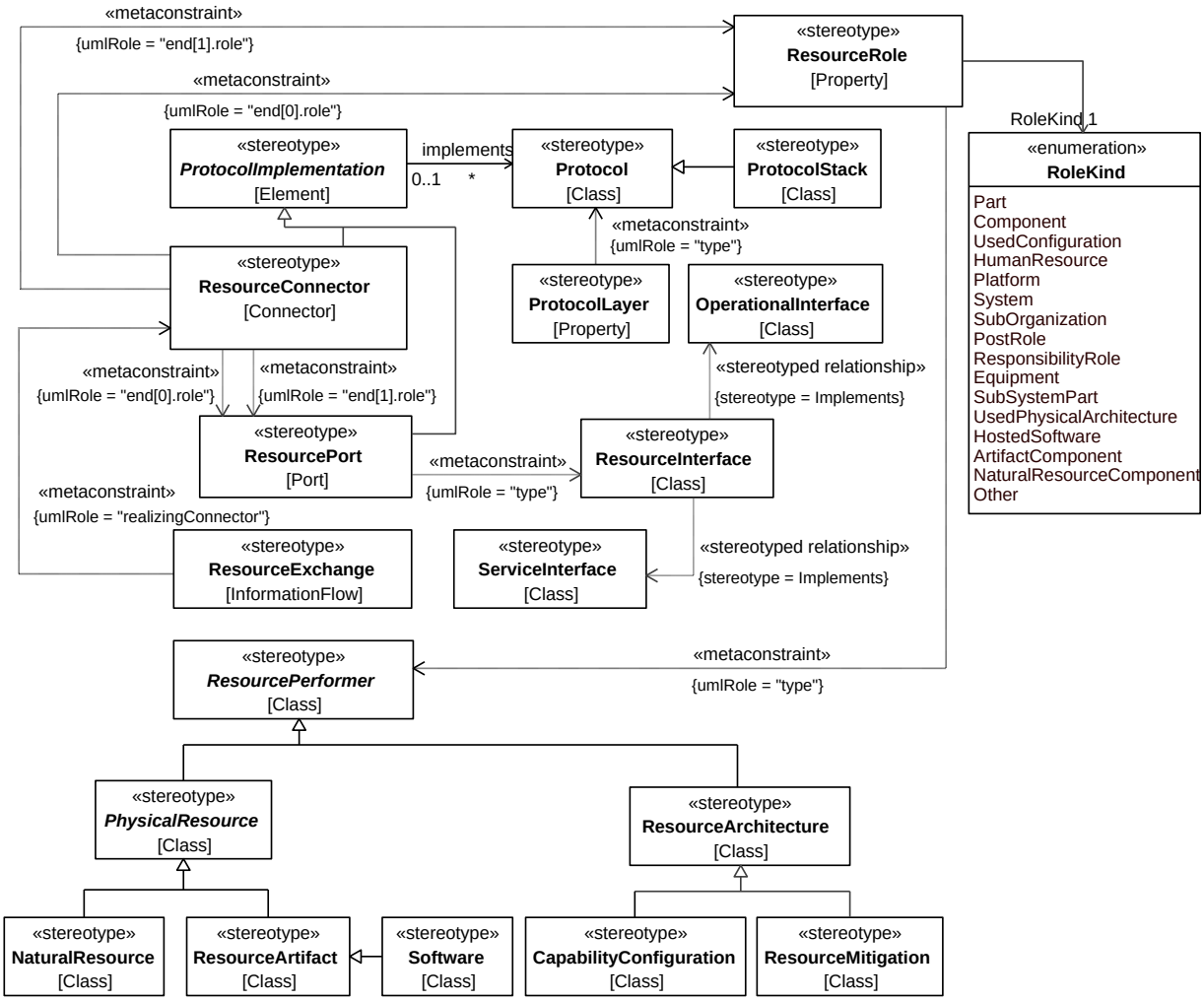

Figure 246 – Resources Structure

Elements

- CapabilityConfiguration
- NaturalResource
- OperationalInterface
- PhysicalResource
- Protocol
- ProtocolImplementation
- ProtocolLayer
- ProtocolStack
- ResourceArchitecture
- ResourceArtifact
- ResourceConnector
- ResourceExchange
- ResourceInterface
- ResourceMitigation
- ResourcePerformer
- ResourcePort
- ResourceRole
- RoleKind
- ServiceInterface

- Software

View Specifications::Resources::Connectivity

Stakeholders: Systems Engineers, IT Architects, Solution Providers, Implementers.
Concerns: capture the interactions between resources.
Definition: summarizes interactions between resources of information, systems, personnel, natural resources etc. and the functions that produce and consume them. Measurements can optionally be included.
Recommended Implementation: tabular format.

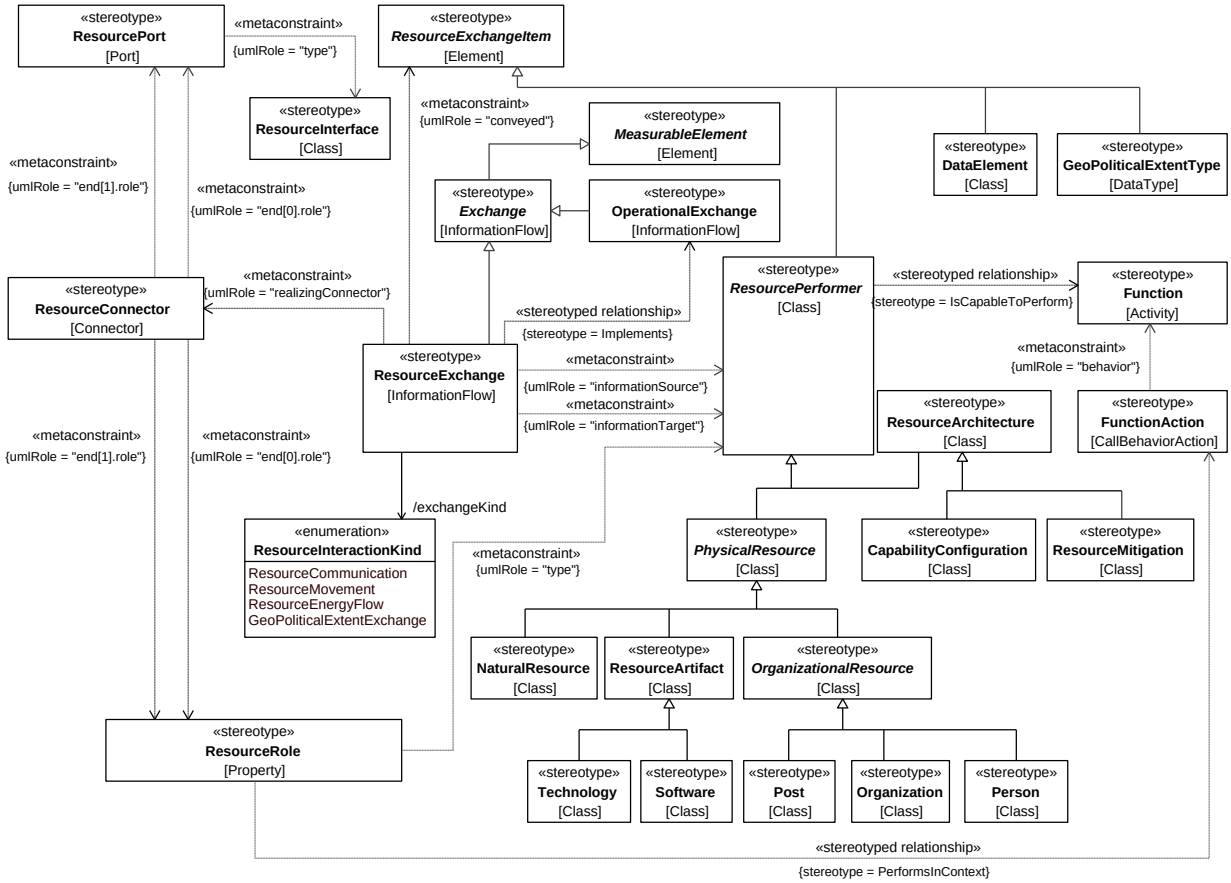

Figure 247 – Resources Connectivity

Elements

- CapabilityConfiguration
- DataElement
- Exchange
- Function
- FunctionAction
- GeoPoliticalExtentType
- MeasurableElement
- NaturalResource
- OperationalExchange
- Organization
- OrganizationalResource
- Person
- PhysicalResource
- Post
- ResourceArchitecture
- ResourceArtifact

- ResourceConnector
- ResourceExchange
- ResourceExchangeItem
- ResourceInteractionKind
- ResourceInterface
- ResourceMitigation
- ResourcePerformer
- ResourcePort
- ResourceRole
- Software
- Technology

View Specifications::Resources::Processes

Stakeholders: Solution Providers, Systems Engineers, IT Architects.
Concerns: captures activity based behavior and flows.
Definition: describes the functions that are normally conducted in the course of implementing operational activity(ies) in support of capability(ies). It describes the functions, their Inputs/Outputs, function actions and flows between them.
Recommended Implementation: SysML Activity Diagram, SysML Block Definition Diagram.

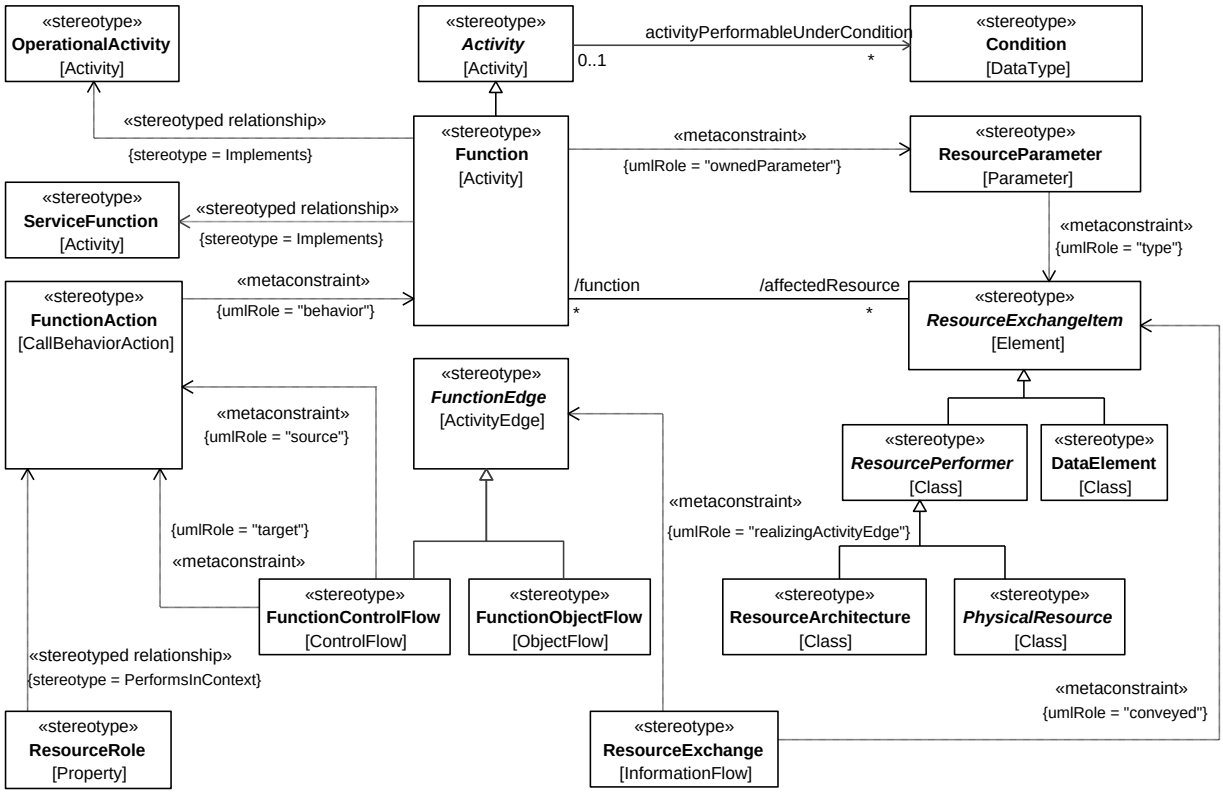

Figure 248 – Resources Processes

Elements

- Activity
- Condition
- DataElement
- Function
- FunctionAction
- FunctionControlFlow
- FunctionEdge
- FunctionObjectFlow

- OperationalActivity
- PhysicalResource
- ResourceArchitecture
- ResourceExchange
- ResourceExchangeItem
- ResourceParameter
- ResourcePerformer
- ResourceRole
- ServiceFunction

View Specifications::Resources::States

Stakeholders: Systems Engineers, Software Engineers.
Concerns: capture state-based behavior of a resource.
Definition: it is a graphical representation of states of a resource and how that resource responds to various events and actions.
Recommended Implementation: SysML State Diagram.

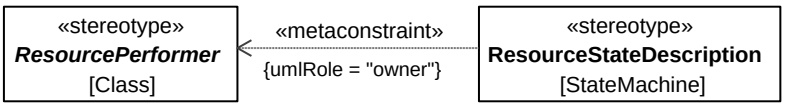

Figure 249 – Resources States

Elements

- ResourcePerformer
- ResourceStateDescription

View Specifications::Resources::Interaction Scenarios

Stakeholders: Software Engineers, Systems Engineers.
Concerns: interactions between resources (roles).
Definition: provides a time-ordered examination of the interactions between resources.
Recommended Implementation: SysML Sequence Diagram.

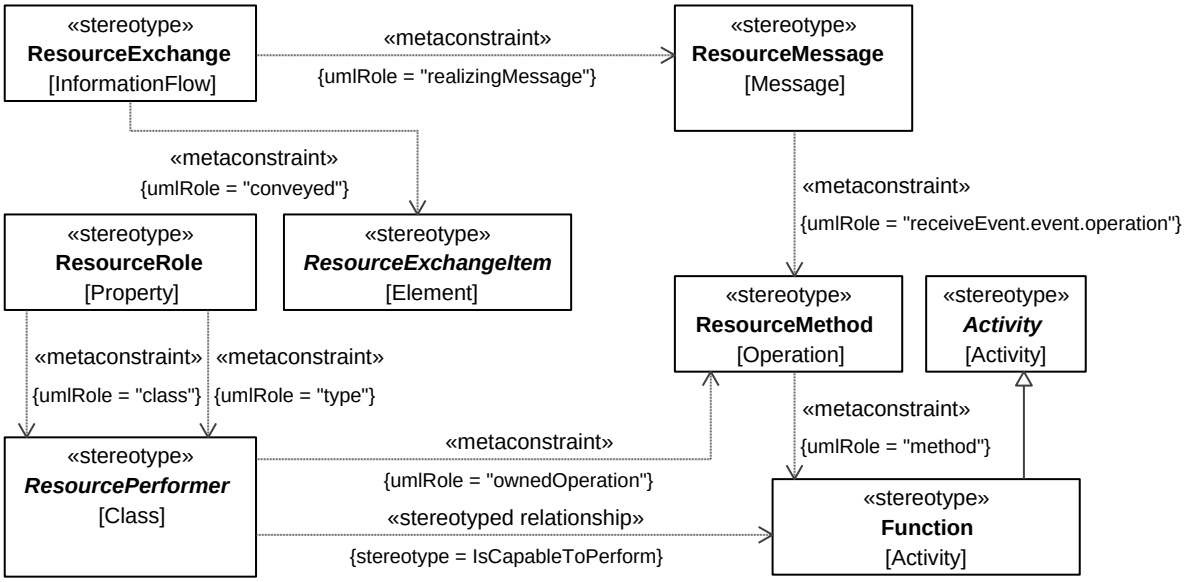

Figure 250 – Resources Interaction Scenarios

Elements

- Activity
- Function
- ResourceExchange

- ResourceExchangeItem
- ResourceMessage
- ResourceMethod
- ResourcePerformer
- ResourceRole

View Specifications::Resources::Constraints

Stakeholders: Systems Engineers, IT Architects, Solution Providers, Implementers.
Concerns: define limitations, constraints and performance parameters for resources, their interactions, performed functions, and data.
Definition: specifies traditional textual rules/non-functional requirements that are constraints on resources, their interactions, performed functions, and data. The addition of SysML parametrics provide a computational means of defining resource constraints within a specific context.
Recommended Implementation: tabular format, SysML Block Definition Diagram, SysML Parametric Diagram, OCL.

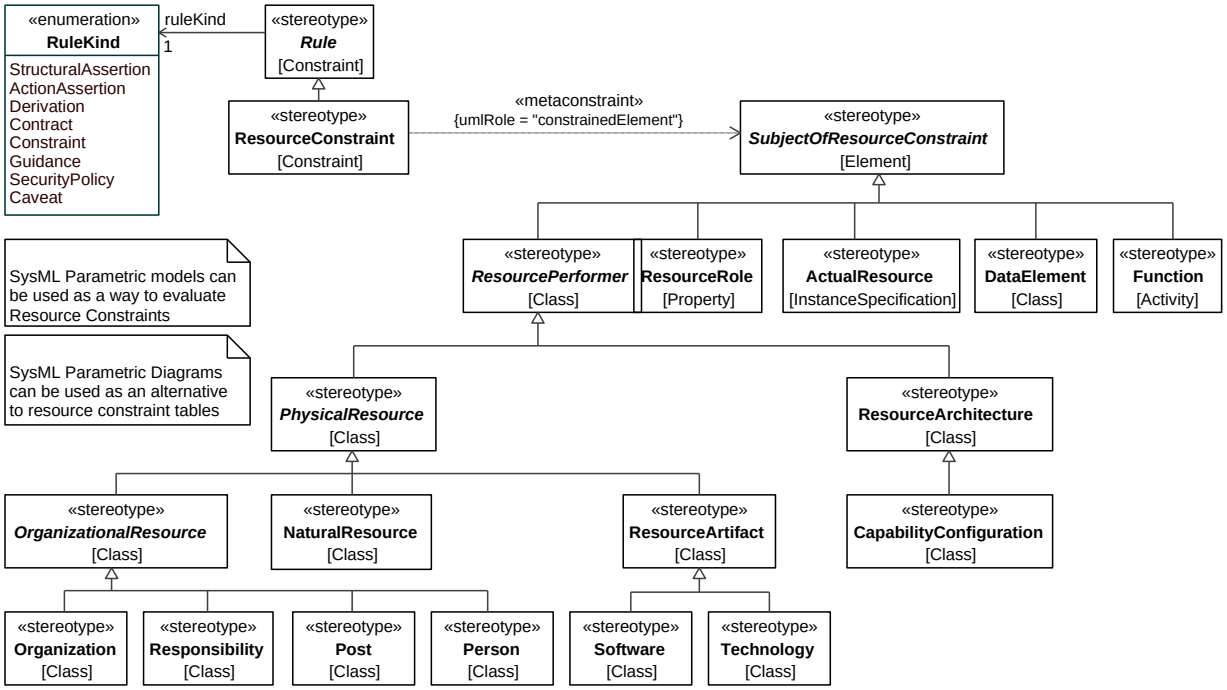

Figure 251 – Resources Constraints

Elements

- ActualResource
- CapabilityConfiguration
- DataElement
- Function
- NaturalResource
- Organization
- OrganizationalResource
- Person
- PhysicalResource
- Post
- ResourceArchitecture
- ResourceArtifact
- ResourceConstraint
- ResourcePerformer
- ResourceRole
- Responsibility

- Rule
- RuleKind
- Software
- SubjectOfResourceConstraint
- Technology

View Specifications::Resources::Roadmap::Resource Roadmap:Evolution

Stakeholders: Systems Engineers, IT Architects, Solution Providers, Implements.
Concerns: resource structure changes over time.
Definition: provides an overview of how a resource structure changes over time. It shows the structure of several resources mapped against a timeline.
Recommended Implementation: timeline, SysML Block Definition Diagram, SysML Internal Block Diagram.

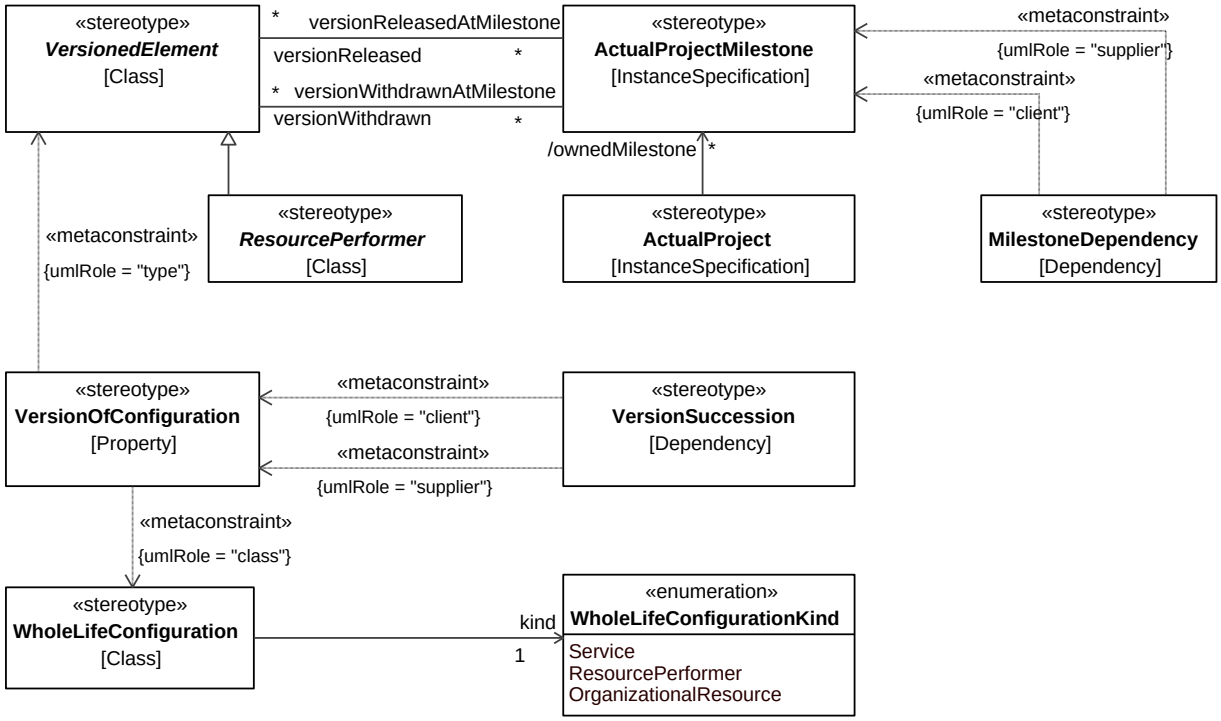

Figure 252 – Resources Roadmap: Evolution

Elements

- ActualProject
- ActualProjectMilestone
- MilestoneDependency
- ResourcePerformer
- VersionedElement
- VersionOfConfiguration
- VersionSuccession
- WholeLifeConfiguration
- WholeLifeConfigurationKind

View Specifications::Resources::Roadmap::Resource Roadmap:Forecast

Stakeholders: Solution Providers, Systems Engineers, IT Architects.
Concerns: technology forecast.
Definition: defines the underlying current and expected supporting technologies. Expected supporting technologies are

those that can be reasonably forecast given the current state of technology, and expected improvements / trends.
Recommended Implementation: timeline, tabular format, SysML Block Definition Diagram.

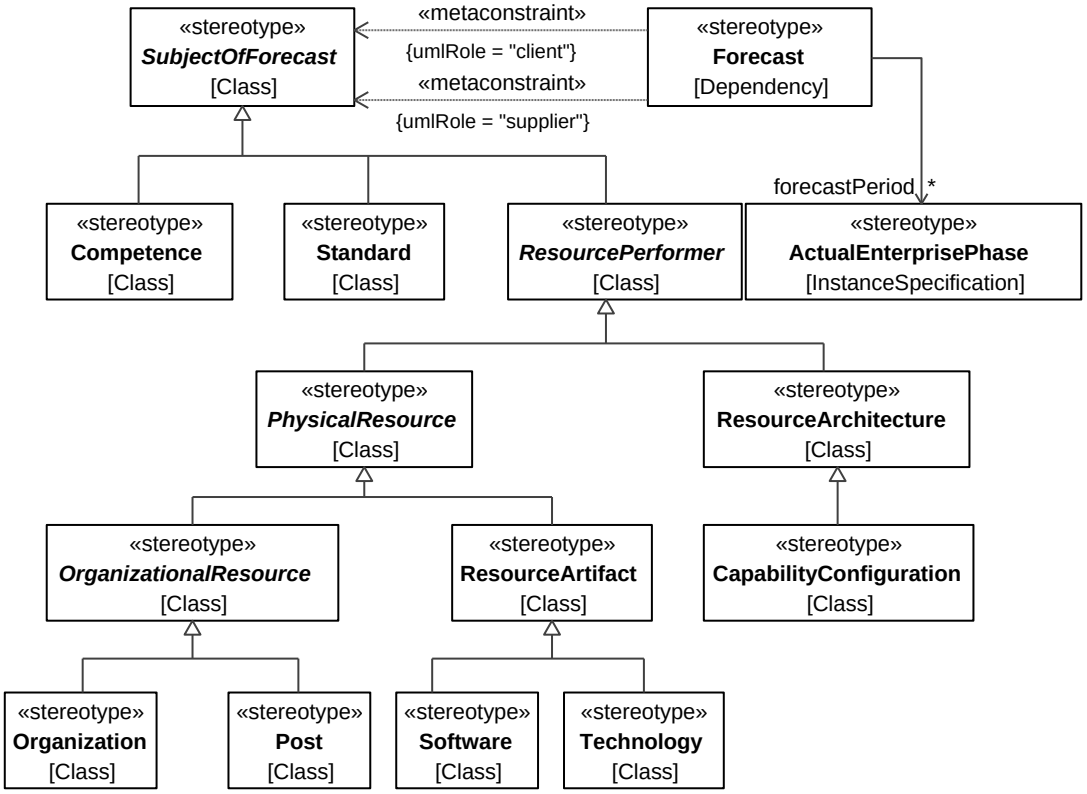

Figure 253 – Resources Roadmap: Forecast

Elements

- ActualEnterprisePhase
- CapabilityConfiguration
- Competence
- Forecast
- Organization
- OrganizationalResource
- PhysicalResource
- Post
- ResourceArchitecture
- ResourceArtifact
- ResourcePerformer
- Software
- Standard
- SubjectOfForecast
- Technology

View Specifications::Resources::Traceability

Stakeholders: Systems Engineers, Enterprise Architects, Solution Providers, Business Architects.
Concerns: traceability between operational activities and functions that implements them.
Definition: depicts the mapping of functions to operational activities and thus identifies the transformation of an operational need into a purposeful function performed by a resource or solution.
Recommended Implementation: Matrix format, SysML Block Definition Diagram.

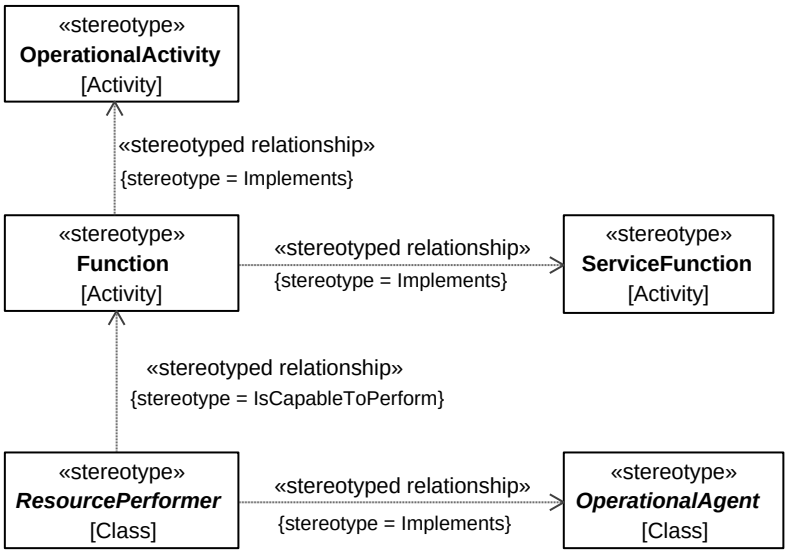

Figure 254 – Resources Traceability

Elements

- Function
- OperationalActivity
- OperationalAgent
- ResourcePerformer
- ServiceFunction

View Specifications::Security

Stakeholders: Security Architects, Security Engineers. Systems Engineers, Operational Architects.
Concerns: addresses the security constraints and information assurance attributes that exist on exchanges between resources and OperationalPerformers
Definition: illustrates the security assets, security constraints, security controls, families, and measures required to address specific security concerns.

View Specifications::Security::Taxonomy

Stakeholders: Security Architects, Security Engineers. Systems Engineers, Operational Architects.
Concerns: Security assets and security enclaves.
Definition: Defines the hierarchy of security assets and asset owners that are available to implement security, security constraints (policy, guidance, laws and regulations) and details where they are located (security enclaves).
Recommended Implementation: SysML Internal Block Diagram, SysML Block Definition Diagram.

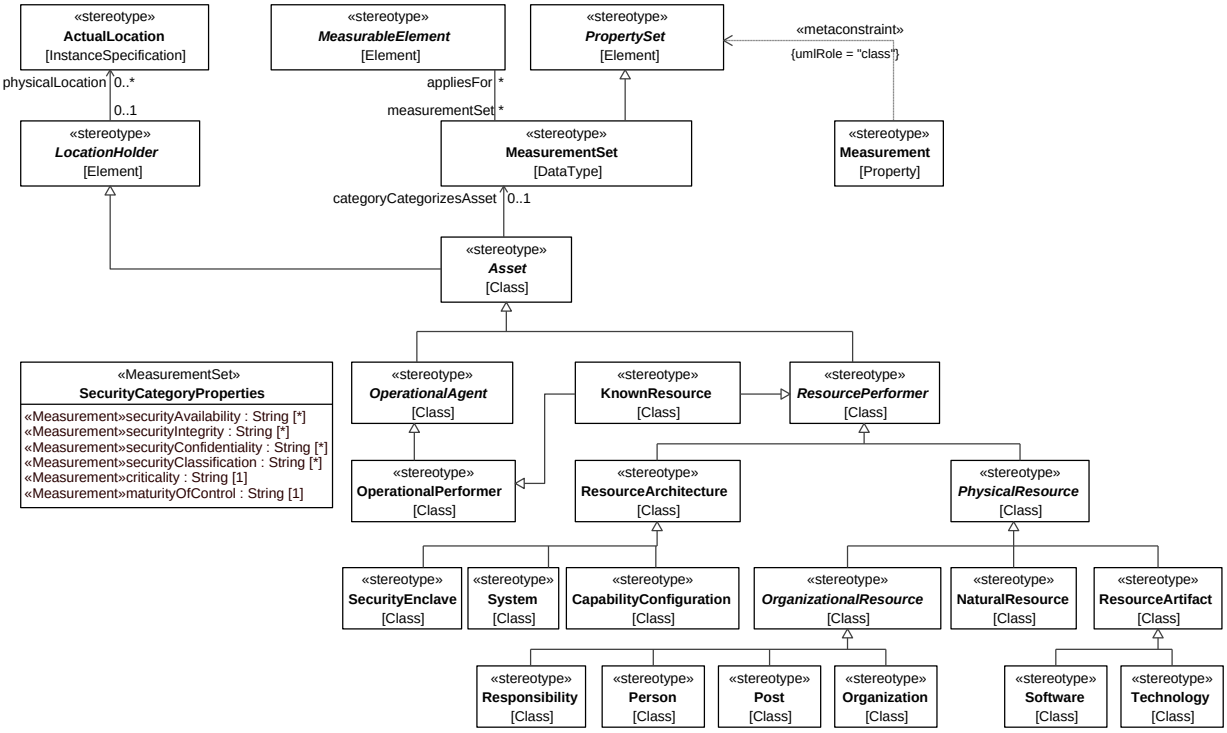

Figure 255 – Security Taxonomy

Elements

- ActualLocation
- Asset
- CapabilityConfiguration
- KnownResource
- LocationHolder
- MeasurableElement
- Measurement
- MeasurementSet
- NaturalResource
- OperationalAgent
- OperationalPerformer
- Organization
- OrganizationalResource
- Person
- PhysicalResource
- Post
- PropertySet
- ResourceArchitecture
- ResourceArtifact
- ResourcePerformer
- Responsibility
- SecurityCategoryProperties
- SecurityEnclave
- Software
- System
- Technology

View Specifications::Security::Structure

Stakeholders: Security Architects, Security Engineers. Systems Engineers, Operational Architects.
Concerns: The structure of security information and where it is used at the operational and resource level.
Definition: Captures the allocation of assets (operational and resource, information and data) across the security enclaves, shows applicable security controls necessary to protect organizations, systems and information during processing, while in storage (bdd), and during transmission (flows on an ibd). This view also captures Asset Aggregation and allocates the usage of the aggregated information at a location through the use of the SecurityProperty.
Recommended Implementation: SysML Internal Block Diagram, SysML Block Definition Diagram.

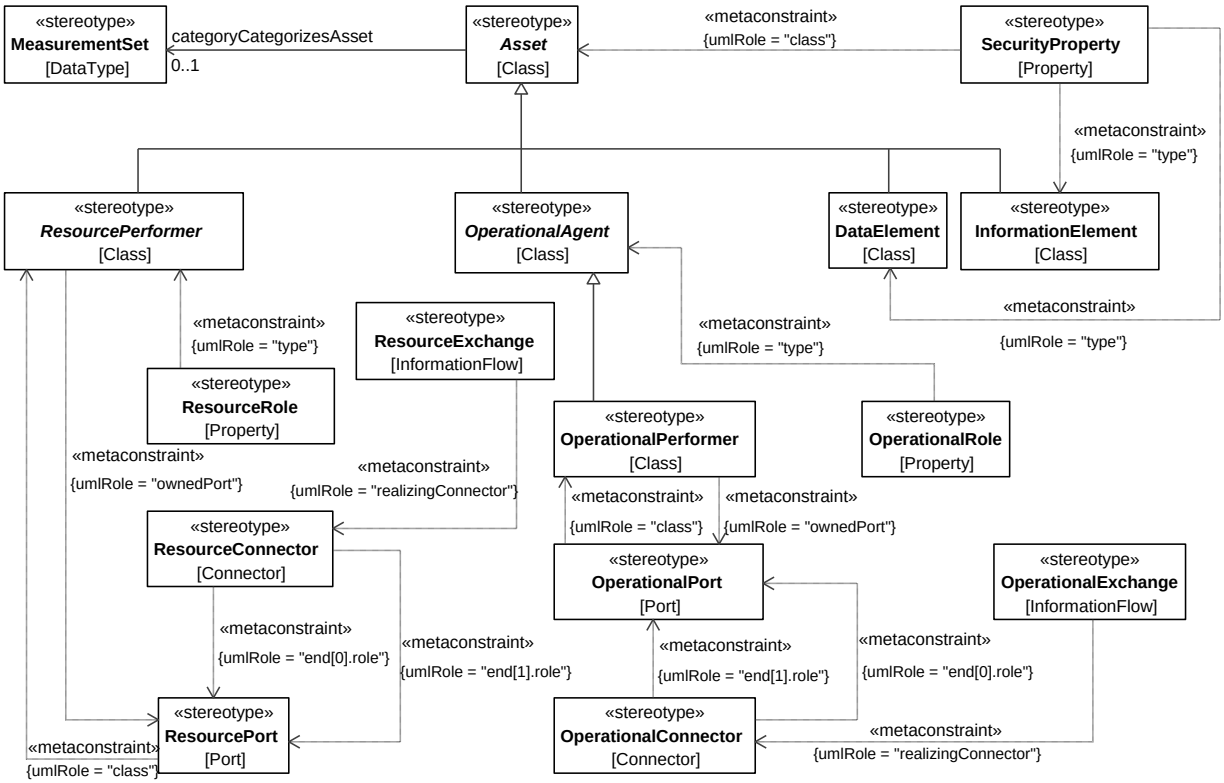

Figure 256 – Security Structure

Elements

- Asset
- DataElement
- InformationElement
- MeasurementSet
- OperationalAgent
- OperationalConnector
- OperationalExchange
- OperationalPerformer
- OperationalPort
- OperationalRole
- ResourceConnector
- ResourceExchange
- ResourcePerformer
- ResourcePort
- ResourceRole
- SecurityProperty

View Specifications::Security::Connectivity

Stakeholders: Security Architects, Security Engineers.
Concerns: Addresses the security constraints and information assurance attributes that exist on exchanges across resources and across performers.
Definition: Lists security exchanges across security assets; the applicable security controls; and the security enclaves that house the producers and consumers of the exchanges. Measurements can optionally be included.
Recommended Implementation: tabular format.

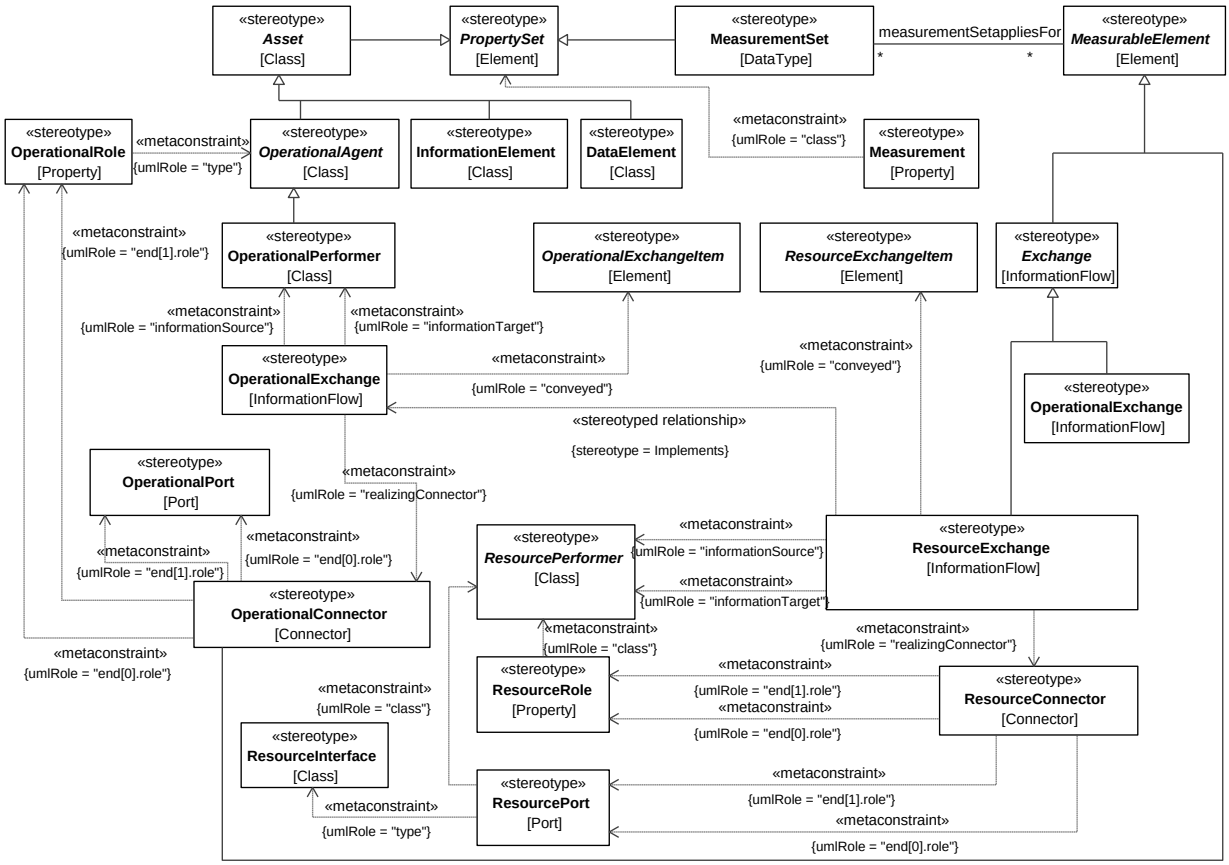

Figure 257 – Security Connectivity

Elements

- Asset
- DataElement
- Exchange
- InformationElement
- MeasurableElement
- Measurement
- MeasurementSet
- OperationalAgent
- OperationalConnector
- OperationalExchange
- OperationalExchangeItem
- OperationalPerformer
- OperationalPort
- OperationalRole
- PropertySet
- ResourceConnector
- ResourceExchange

- ResourceExchangeItem
- ResourceInterface
- ResourcePerformer
- ResourcePort
- ResourceRole

View Specifications::Security::Processes

Stakeholders: Security Architects, Security Engineers.
Concerns: The specification of the Security Control families, security controls, and measures required to address a specific security baseline.
Definition: Provides a set of Security Controls and any possible enhancements as applicable to assets. The activity diagram describes operational or resource level processes that apply (operational level) or implement (resource level) security controls/enhancements to assets located in enclaves and across enclaves. This Security Process view can be instantiated either as a variant of an activity/flow diagram or as a hierarchical work breakdown structure.
Recommended Implementation: SysML Activity Diagram, SysML Block Definition Diagram.

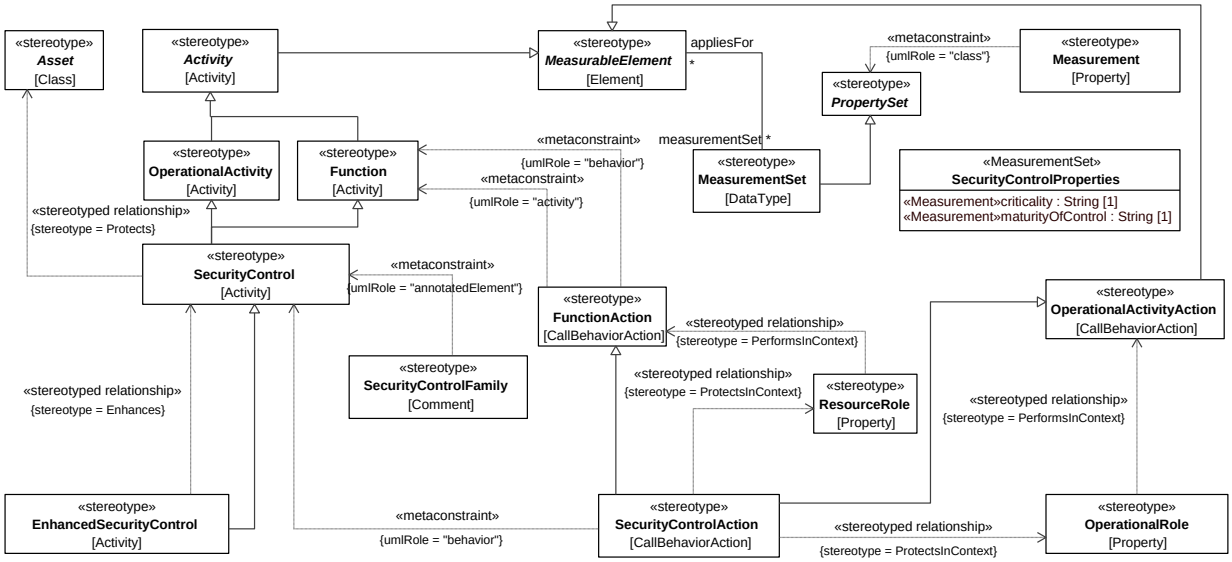

Figure 258 – Security Processes

Elements

- Activity
- Asset
- EnhancedSecurityControl
- Function
- FunctionAction
- MeasurableElement
- Measurement
- MeasurementSet
- OperationalActivity
- OperationalActivityAction
- OperationalRole
- PropertySet
- ResourceRole
- SecurityControl
- SecurityControlAction
- SecurityControlFamily
- SecurityControlProperties

View Specifications::Security::Constraints

Stakeholders: Security Architects, Security Engineers, Risk Analysts.
Concerns: (i) Security-related policy, guidance, laws and regulations as applicable to assets, (ii) threats, vulnerabilities, and risk assessments as applicable to assets.
Definition: (i) Specifies textual rules/non-functional requirements that are security constraints on resources, information and data (e.g. security-related in the form of rules (e.g. access control policy). A common way of representing access control policy is through the use of XACML (eXtensible Access Control Markup Language), it is expected that implementations of UAF allow users to link security constraints to external files represented in XACML. (ii) Identifies risks, specifies risk likelihood, impact, asset criticality, other measurements and enables risk assessment.
Recommended Implementation: tabular or Matrix format, SysML Block Definition Diagram, SysML Parametric Diagram, or OCL.

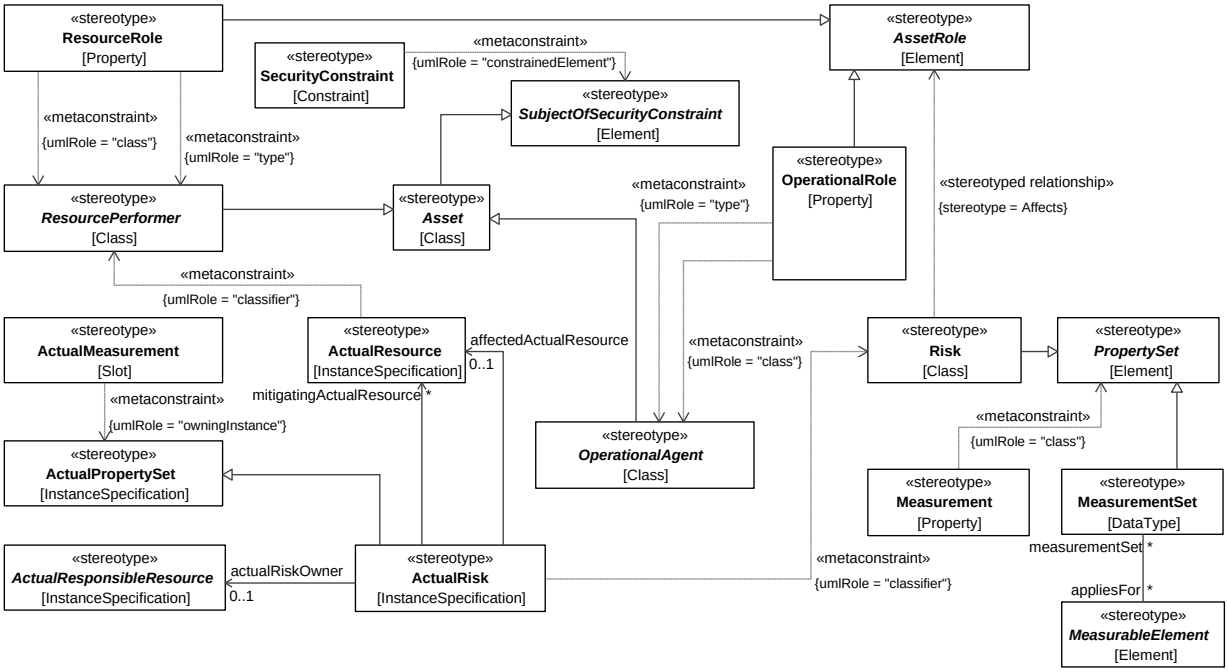

Figure 259 – Security Constraints

Elements

- ActualMeasurement
- ActualPropertySet
- ActualResource
- ActualResponsibleResource
- ActualRisk
- Asset
- AssetRole
- MeasurableElement
- Measurement
- MeasurementSet
- OperationalAgent
- OperationalRole
- PropertySet
- ResourcePerformer
- ResourceRole
- Risk
- SecurityConstraint
- SubjectOfSecurityConstraint

View Specifications::Security::Traceability

Stakeholders: Security Architects, Security Engineers, Risk Analysts.
Concerns: traceability between risk and risk owner, risk mitigations, and affected asset roles.
Definition: depicts the mapping of a risk to each of the following: risk owner, risk mitigations, and affected asset roles.
Recommended Implementation: Matrix format, SysML Block Definition Diagram.

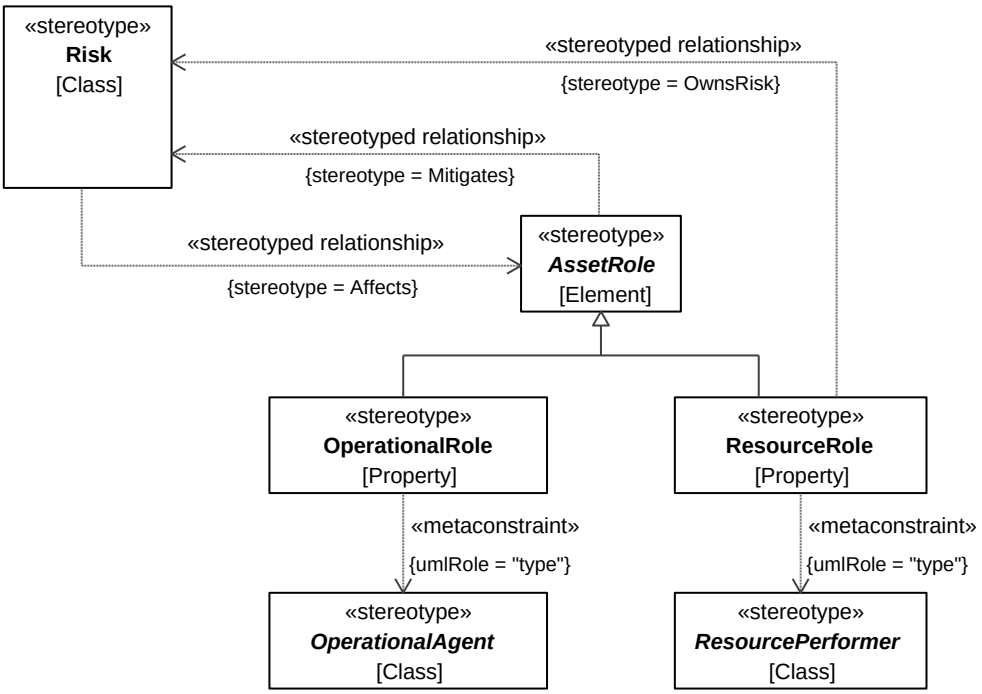

Figure 260 – Security Traceability

Elements

- AssetRole
- OperationalAgent
- OperationalRole
- ResourcePerformer
- ResourceRole
- Risk

View Specifications::Projects

Stakeholders: PMs, Project Portfolio Managers, Enterprise Architects.
Concerns: project portfolio, projects and project milestones.
Definition: describes projects and project milestones, how those projects deliver capabilities, the organizations contributing to the projects and dependencies between projects.

View Specifications::Projects::Taxonomy

Stakeholders: PMs, Project Portfolio Managers, Enterprise Architects.
Concerns: types of projects and project milestones.
Definition: shows the taxonomy of types of projects and project milestones.
Recommended Implementation: SysML Block Definition Diagram.

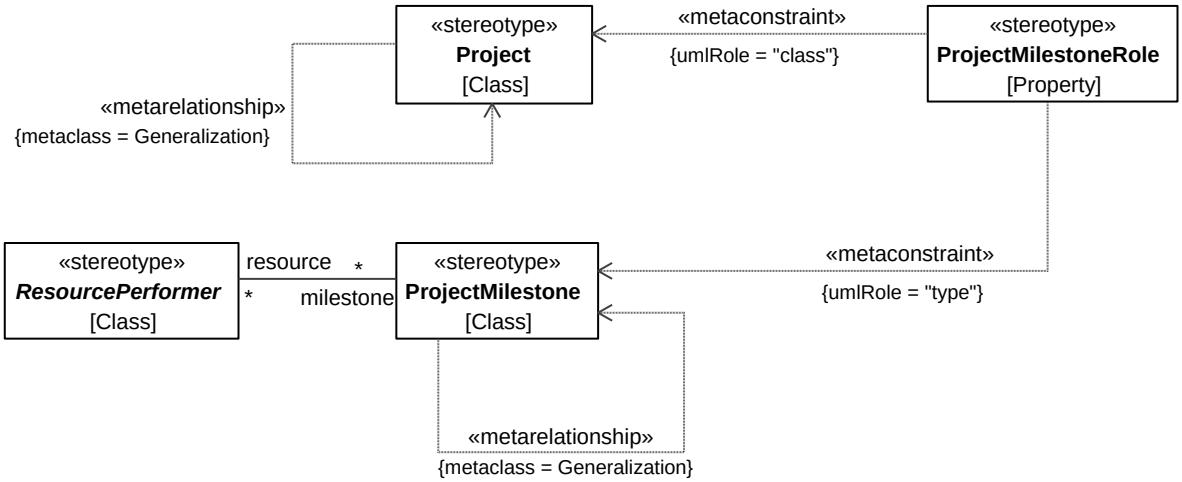

Figure 261 – Project Taxonomy

Elements

- Project
- ProjectMilestone
- ProjectMilestoneRole
- ResourcePerformer

View Specifications::Projects::Structure

Stakeholders: PMs.
Concerns: relationships between types of projects and project milestones.
Definition: provides a template for an actual project(s) road map(s) to be implemented.
Recommended Implementation: SysML Block Definition Diagram.

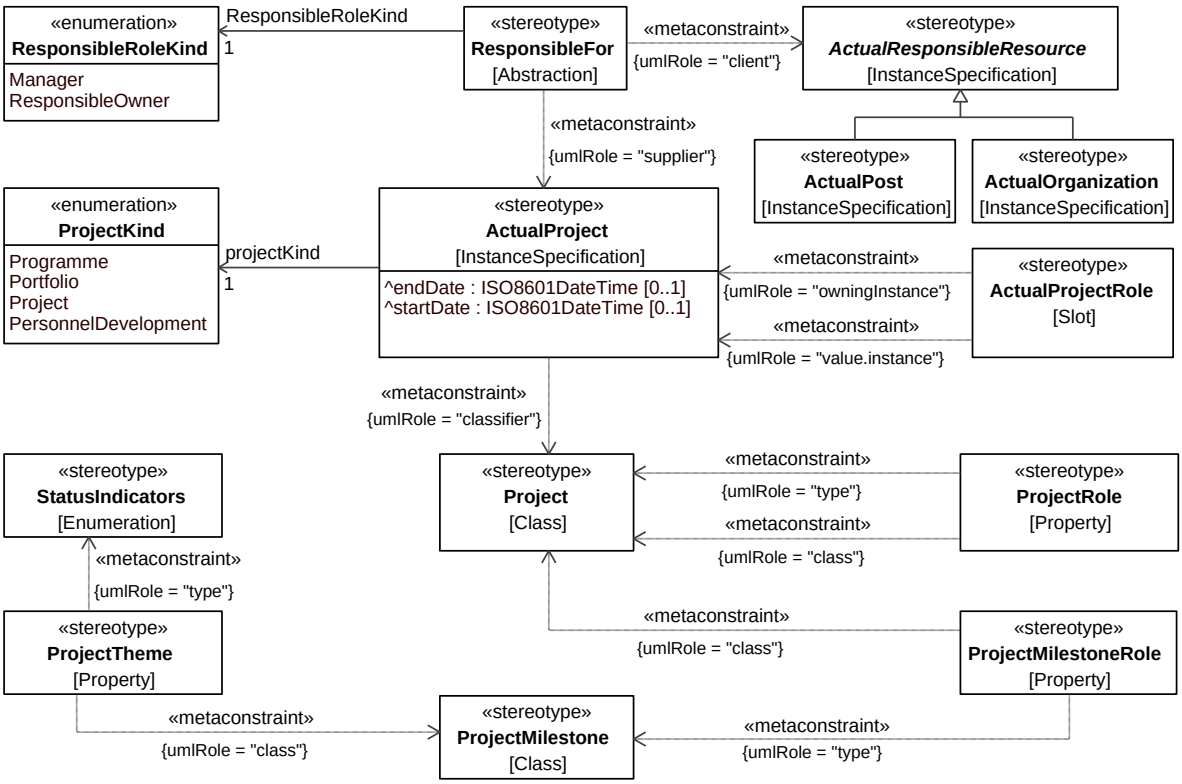

Figure 262 – Project Structure

Elements

- ActualOrganization
- ActualPost
- ActualProject
- ActualProjectRole
- ActualResponsibleResource
- Project
- ProjectKind
- ProjectMilestone
- ProjectMilestoneRole
- ProjectRole
- ProjectTheme
- ResponsibleFor
- ResponsibleRoleKind
- StatusIndicators

View Specifications::Projects::Connectivity

Stakeholders: PMs.
Concerns: relationships between projects and project milestones.
Definition: shows how projects and project milestones are related in sequence.
Recommended Implementation: SysML Block Definition Diagram.

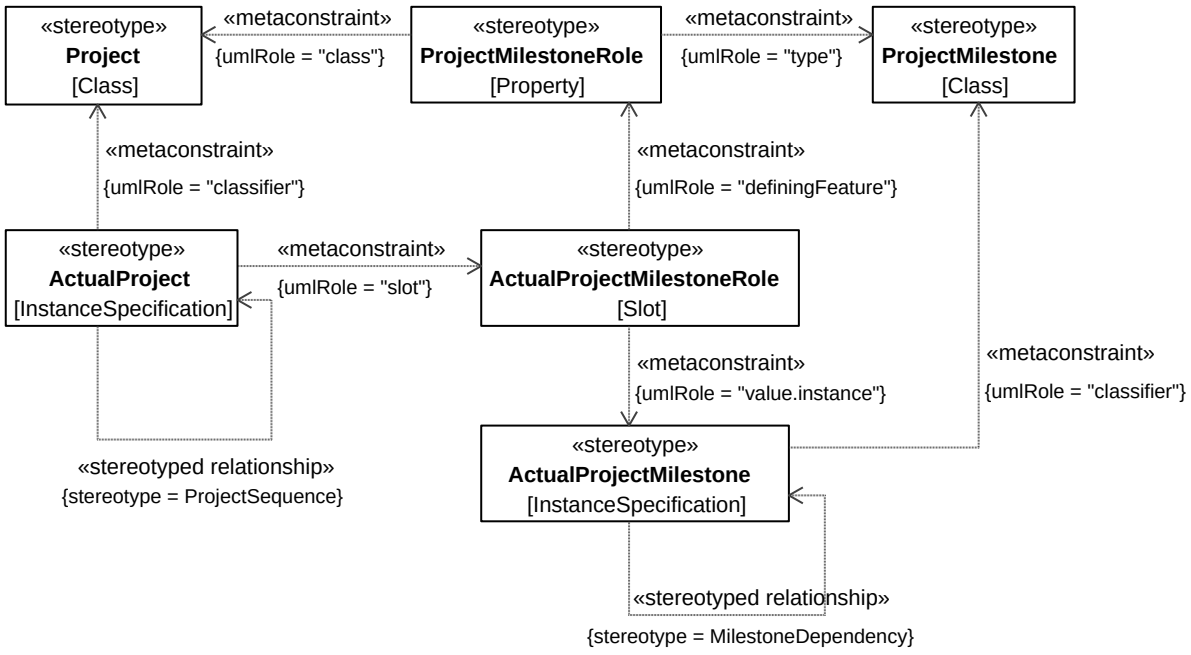

Figure 263 – Project Connectivity

Elements

- ActualProject
- ActualProjectMilestone
- ActualProjectMilestoneRole
- Project
- ProjectMilestone
- ProjectMilestoneRole

View Specifications::Projects::Roadmap

Stakeholders: PMs, Capability Owners, Solution Providers, Enterprise Architects.
Concerns: the product portfolio management; a planning of capability delivery.
Definition: provides a timeline perspective on programs or projects
Recommended Implementation: timeline, tabular format, SysML Block Definition Diagram.

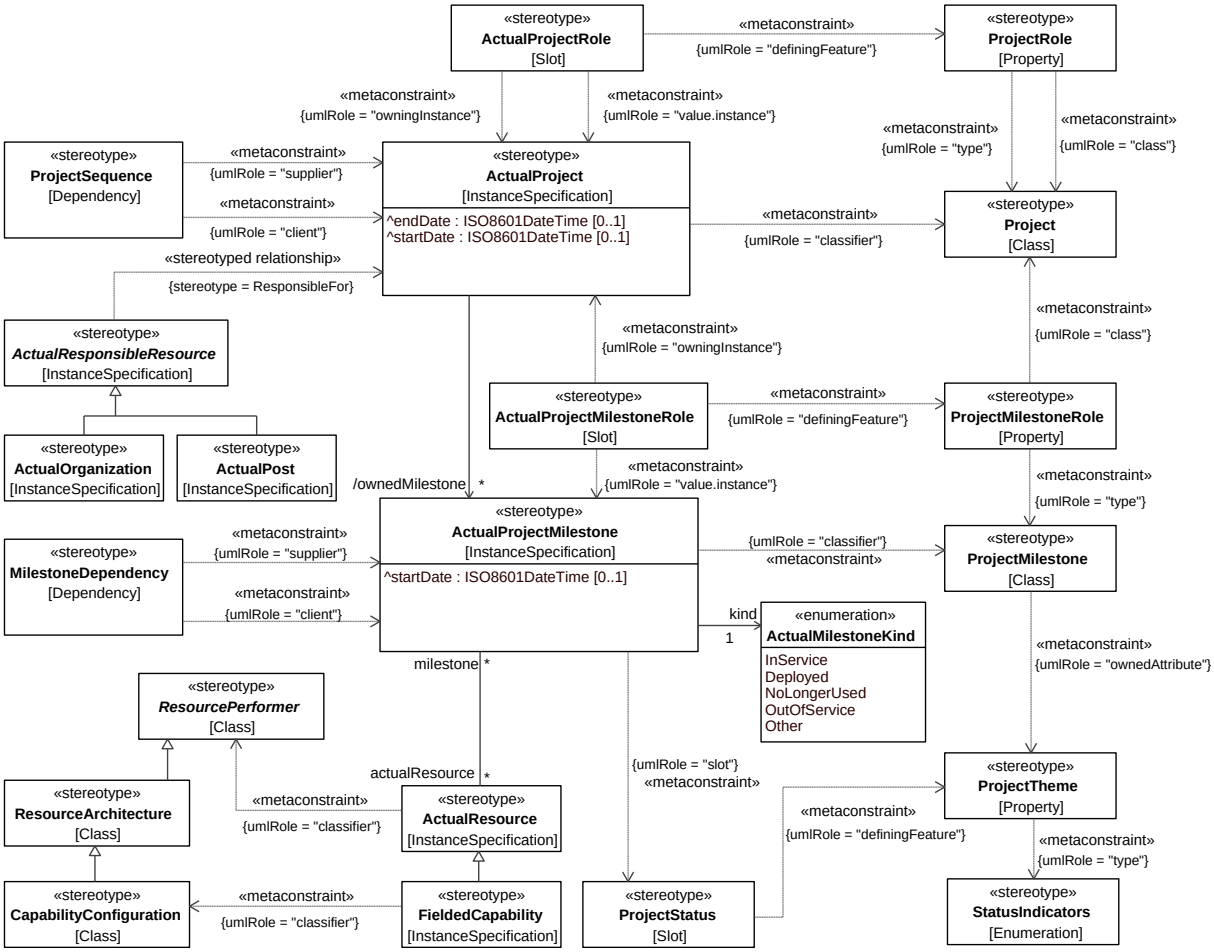

Figure 264 – Project Roadmap

Elements

- ActualMilestoneKind
- ActualOrganization
- ActualPost
- ActualProject
- ActualProjectMilestone
- ActualProjectMilestoneRole
- ActualProjectRole
- ActualResource
- ActualResponsibleResource
- CapabilityConfiguration
- FieldedCapability
- MilestoneDependency
- Project
- ProjectMilestone
- ProjectMilestoneRole
- ProjectRole

- ProjectSequence
- ProjectStatus
- ProjectTheme
- ResourceArchitecture
- ResourcePerformer
- StatusIndicators

View Specifications::Projects::Traceability

Stakeholders: PMs, Project Portfolio Managers, Enterprise Architects.
Concerns: traceability between capabilities and projects that deliver them.
Definition: depicts the mapping of projects to capabilities and thus identifies the transformation of a capability(ies) into a purposeful implementation via projects.
Recommended Implementation: Matrix format, SysML Block Definition Diagram.

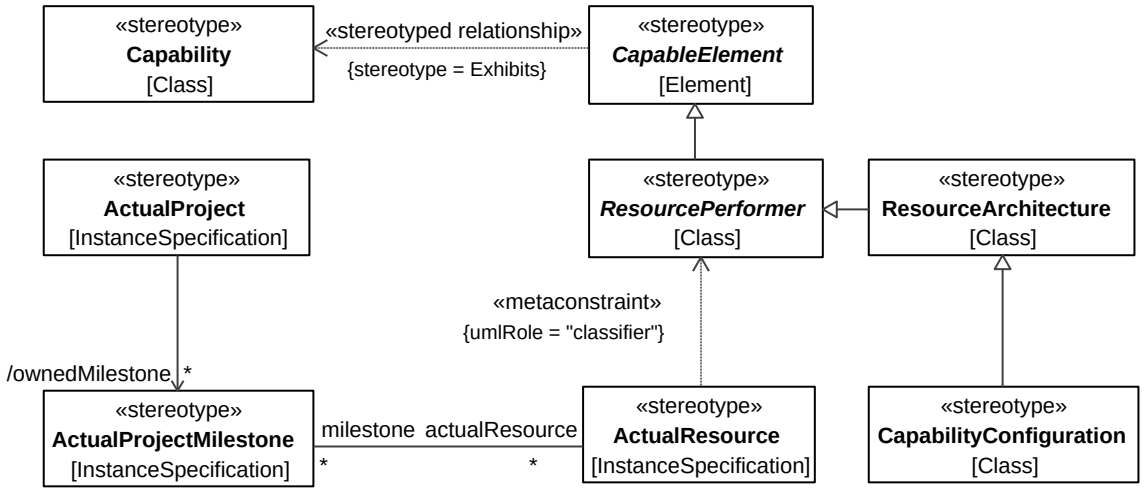

Figure 265 – Project Traceability

Elements

- ActualProject
- ActualProjectMilestone
- ActualResource
- Capability
- CapabilityConfiguration
- CapableElement
- ResourceArchitecture
- ResourcePerformer

View Specifications::Standards

Stakeholders: Solution Providers, Systems Engineers, Software Engineers, Systems Architects, Business Architects.
Concerns: technical and non-technical Standards applicable to the architecture.
Definition: shows the technical, operational, and business Standards applicable to the architecture. Defines the underlying current and expected Standards.

View Specifications::Standards::Taxonomy

Stakeholders: Solution Providers, Systems Engineers, Software Engineers, Systems Architects, Business Architects.
Concerns: technical and non-technical standards, guidance and policy applicable to the architecture.
Definition: shows the taxonomy of types of technical, operational, and business standards, guidance and policy applicable to the architecture.
Recommended Implementation: SysML Block Definition Diagram.

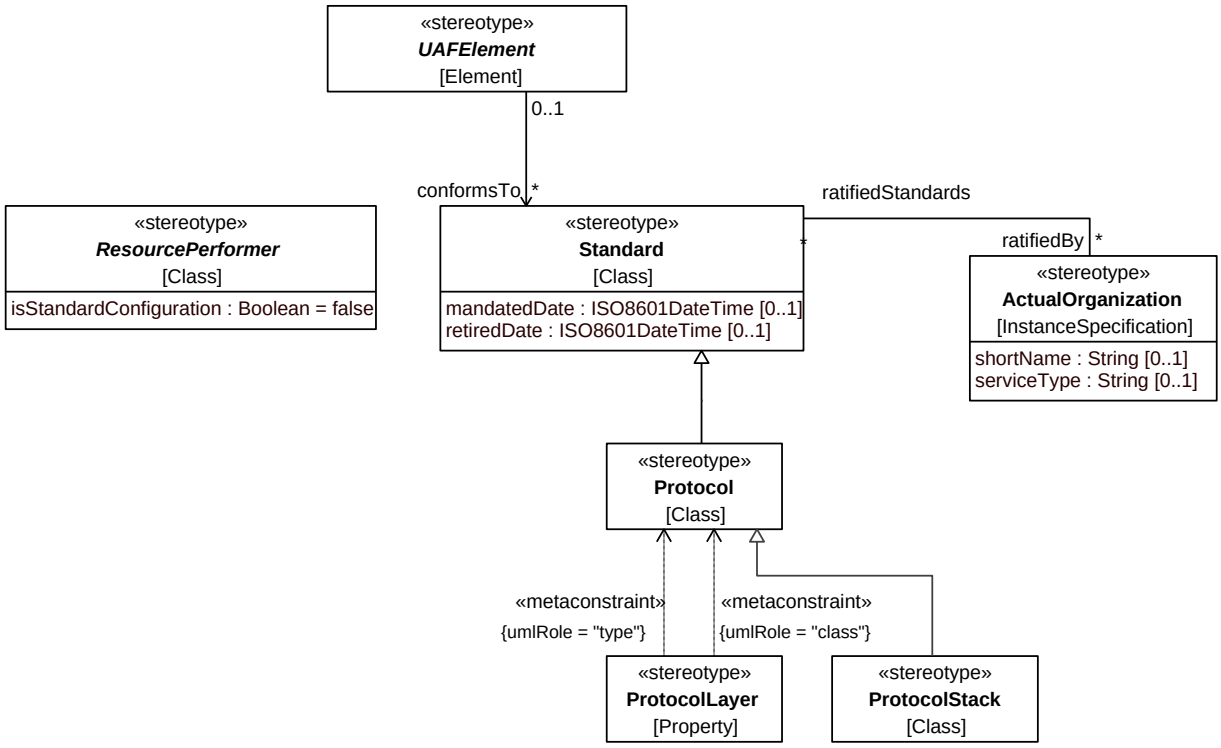

Figure 266 – Standards Taxonomy

Elements

- ActualOrganization
- Protocol
- ProtocolLayer
- ProtocolStack
- ResourcePerformer
- Standard
- UAFElement

View Specifications::Standards::Structure

Stakeholders: Solution Providers, Systems Engineers, Software Engineers, Systems Architects.
Concerns: the specification of the protocol stack used in the architecture.
Definition: shows the composition of standards required to achieve the architecture's objectives.
Recommended Implementation: SysML Internal Block Diagram.

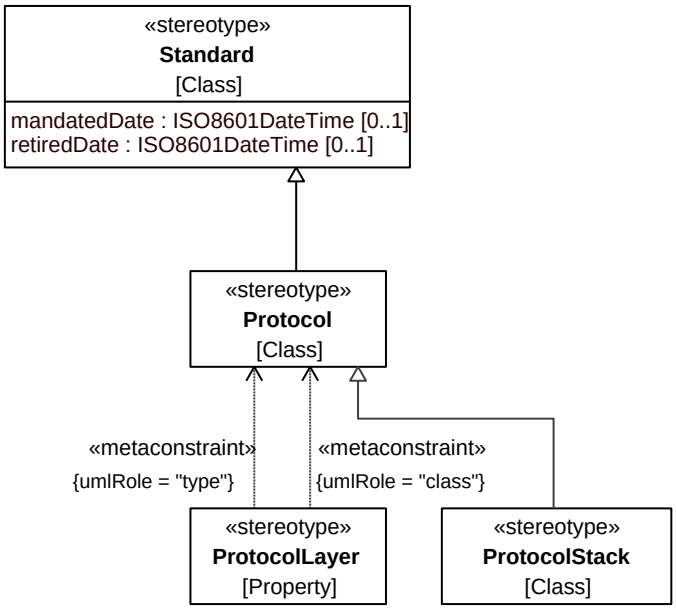

Figure 267 – Standards Structure

Elements

- Protocol
- ProtocolLayer
- ProtocolStack
- Standard

View Specifications::Standards::Roadmap

Stakeholders: Solution Providers, Systems Engineers, Systems Architects, Software Engineers, Business Architects.
Concerns: expected changes in technology-related standards and conventions, operational standards, or business standards and conventions.
Definition: defines the underlying current and expected standards. Expected standards are those that can be reasonably forecast given the current state of technology, and expected improvements / trends.
Recommended Implementation: timeline, tabular format, SysML Block Definition Diagram.

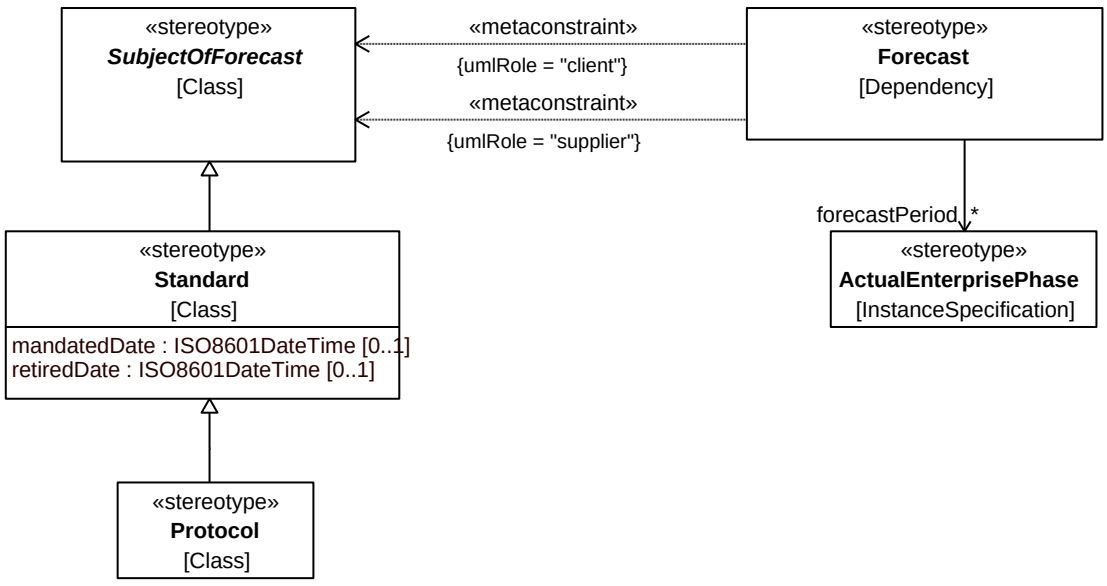

Figure 268 – Standards Roadmap

Elements

- ActualEnterprisePhase
- Forecast
- Protocol
- Standard
- SubjectOfForecast

View Specifications::Standards::Traceability

Stakeholders: Solution Providers, Systems Engineers, Software Engineers, Systems Architects, Business Architects.
Concerns: standards that need to be taken in account to ensure the interoperability of the implementation of architectural elements.
Definition: shows the applicability of standards to specific elements in the architecture.
Recommended Implementation: tabular format, matrix format, SysML Block Definition Diagram.

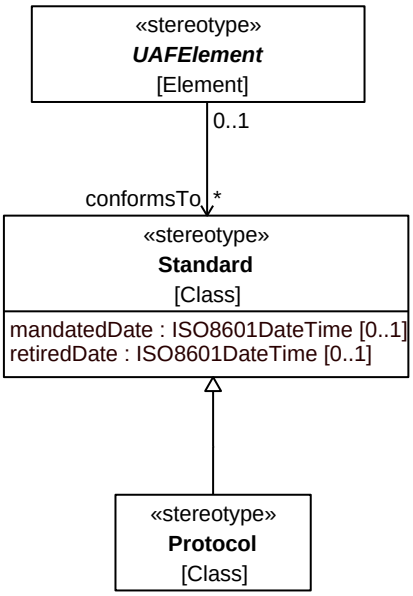

Figure 269 – Standards Traceability

Elements

- Protocol
- Standard
- UAFElement

View Specifications::Actual Resources

Stakeholders: Solution Providers, Systems Engineers, Business Architects, Human Resources.
Concerns: the analysis.e.g. evaluation of different alternatives, what-if, trade-offs, V&V on the actual resource configurations.
Definition: illustrates the expected or achieved actual resource configurations and actual relationships between them.

View Specifications::Actual Resources::Structure

Stakeholders: Solution Providers, Systems Engineers, Business Architects.
Concerns: the analysis, e.g. evaluation of different alternatives, what-if, trade-offs, V&V on the actual resource configurations as it provides a means to capture different solution architectures. The detailed analysis (trade-off, what-if etc.) is carried out using the Resource Constraints view.
Definition: illustrates the expected or achieved actual resource configurations required to meet an operational need.
Recommended Implementation: SysML Block Definition Diagram, SysML Internal Block Diagram.

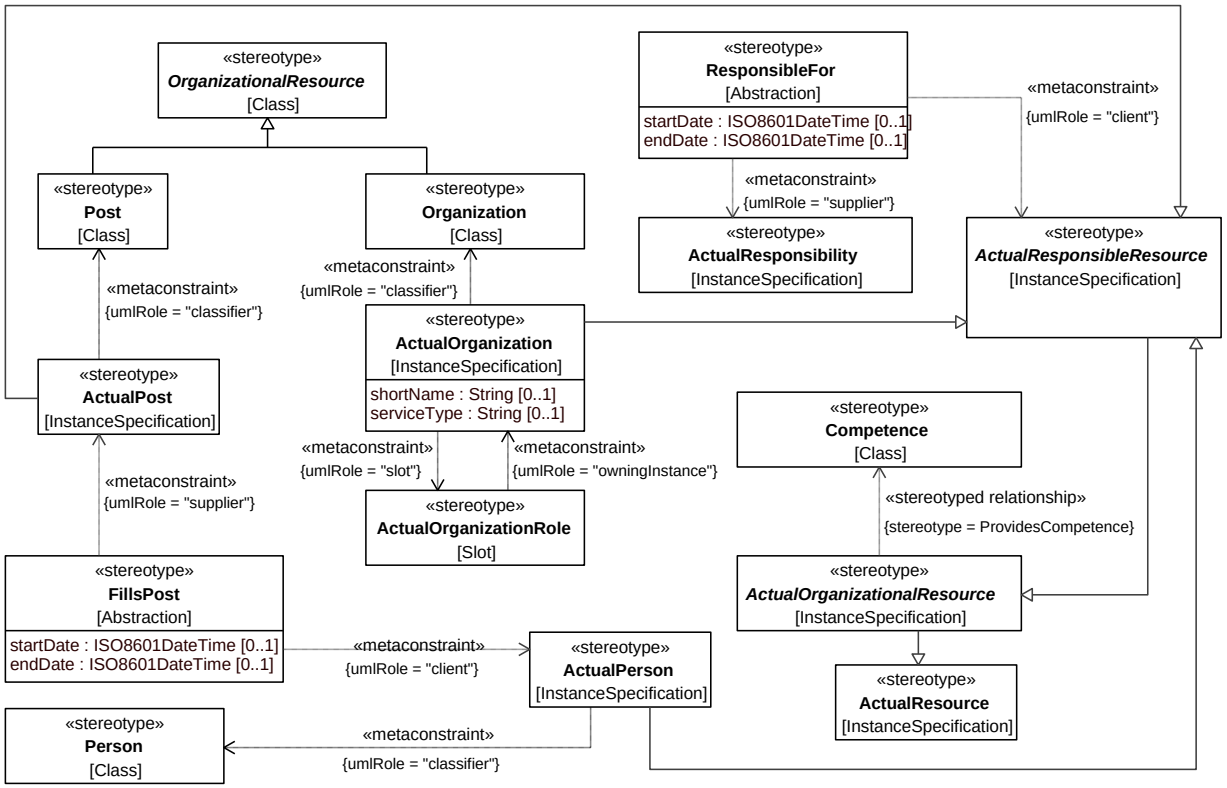

Figure 270 – Actual Resources Structure

Elements

- ActualOrganization
- ActualOrganizationalResource
- ActualOrganizationRole
- ActualPerson
- ActualPost
- ActualResource
- ActualResponsibility
- ActualResponsibleResource
- Competence
- FillsPost
- Organization
- OrganizationalResource
- Person
- Post
- ResponsibleFor

View Specifications::Actual Resources::Connectivity

Stakeholders: Solution Providers, Systems Engineers, Business Architects.
Concerns: the communication of actual resource.
Definition: illustrates the actual resource configurations and actual relationships between them.
Recommended Implementation: tabular format, SysML Block Definition Diagram, SysML Internal Block Diagram, SysML Sequence Diagram.

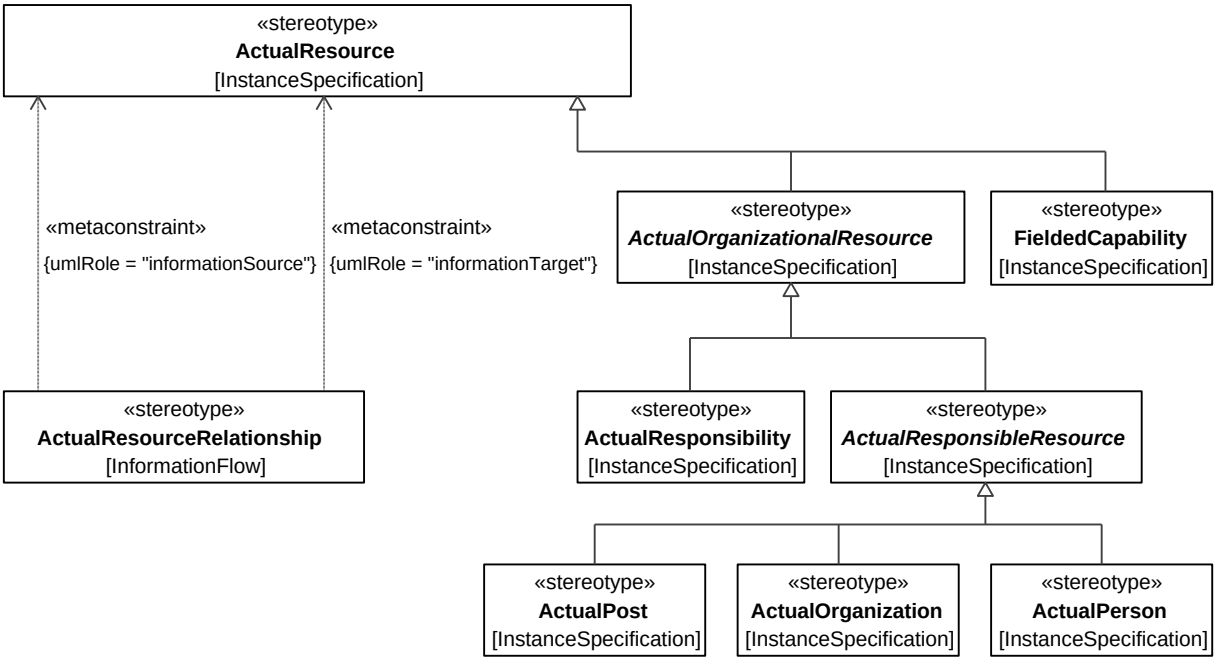

Figure 271 – Actual Resources Connectivity

Elements

- ActualOrganization
- ActualOrganizationalResource
- ActualPerson
- ActualPost
- ActualResource
- ActualResourceRelationship
- ActualResponsibility
- ActualResponsibleResource
- FieldedCapability

View Specifications::Dictionary

Stakeholders: Architects, users of the architecture, Capability Owners, Systems Engineers, Solution Providers.
Concerns: Definitions for all the elements in the architecture, libraries of environments and measurements.
Definition: Presents all the elements used in an architecture. Can be used specifically to capture:
a. elements and relationships that are involved in defining the environments applicable to capability, operational concept or set of systems.
b. measurable properties that can be used to support analysis such as KPIs, MoEs, TPIs etc.

View Specifications::Dictionary::Dictionary

Stakeholders: Solution Providers, Systems Engineers, Software Architects, Business Architects.
Concerns: provides a central reference for a given architecture's data and metadata. It enables the set of architecture description to stand alone, with minimal reference to outside resources.
Definition: contains definitions of terms used in the given architecture. It consists of textual definitions in the form of a glossary, their taxonomies, and their metadata (i.e., data about architecture data), including metadata for any custom-tailored views. Architects should use standard terms where possible (i.e., terms from existing, approved dictionaries, glossaries, and lexicons).
Recommended Implementation: text, table format.

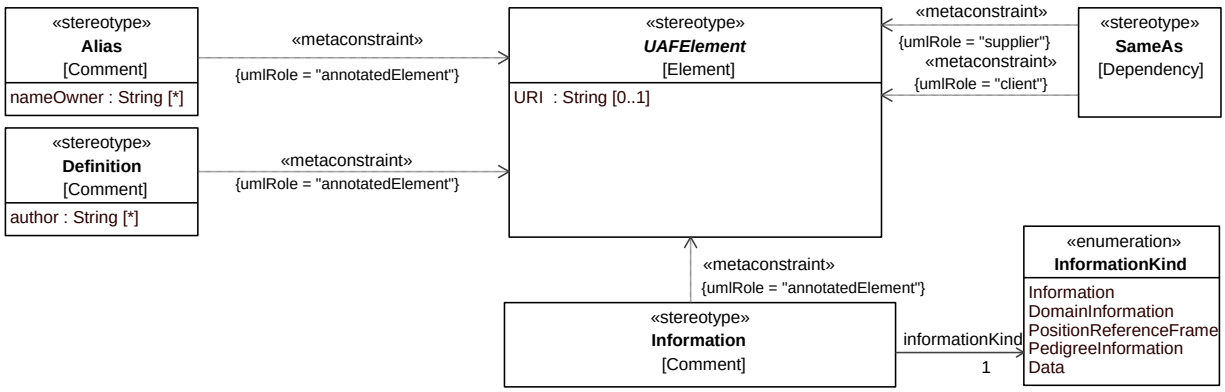

Figure 272 – Dictionary

Elements

- Alias
- Definition
- Information
- InformationKind
- SameAs
- UAFElement

View Specifications::Requirements

Stakeholders: Requirement Engineers, Solution Providers, Systems Engineers, Software Engineers, Systems Architects, Business Architects.
Concerns: requirements traceability.
Definition: used to represent requirements, their properties, and relationships (trace, verify, satisfy, refine) to UAF architectural elements.

View Specifications::Requirements::Requirements

Stakeholders: Requirement Engineers, Solution Providers, Systems Engineers, Software Engineers, Systems Architects, Business Architects.
Concerns: provides a central reference for a set of stakeholder needs expressed as requirements, their relationship (via traceability) to more detailed requirements and the solution described by the architecture that will meet those requirements.
Definition: used to represent requirements, their properties, and relationships (trace, verify, satisfy, refine) between each other and to UAF architectural elements.
Recommended Implementation: SysML Requirement Diagram, tabular format, matrix format.

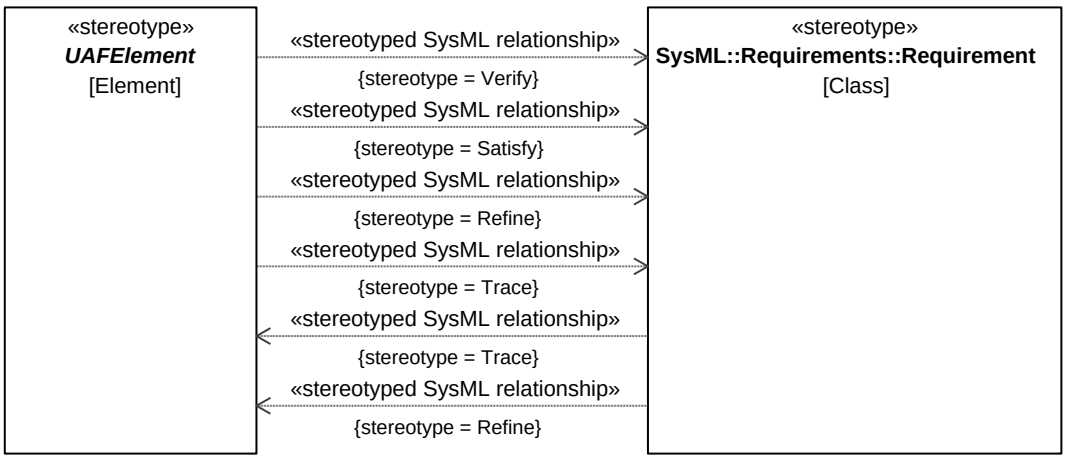

Figure 273 – Requirements

Elements

- Requirement
- UAFElement

View Specifications::Sumary & Overview

Stakeholders: Executives, PMs, Enterprise Architects.
Concerns: executive-level summary information in a consistent form.
Definition: provides executive-level summary information in a consistent form that allows quick reference and comparison between architectural descriptions. Includes assumptions, constraints, and limitations that may affect high-level decisions relating to an architecture-based work programme.

View Specifications::Sumary & Overview::Summary & Overview

Stakeholders: Decision makers, Solution Providers, Systems Engineers, Software Architects, Business Architects.
Concerns: quick overview of an architecture description and summary of analysis. In the initial phases of architecture development, it serves as a planning guide. Upon completion of an architecture, it provides a summary of findings, and any conducted analysis.
Definition: provides executive-level summary information in a consistent form that allows quick reference and comparison among architectures. The Summary and Overview includes assumptions, constraints, and limitations that may affect high-level decision processes involving the architecture.
Recommended Implementation: text, free form diagram, table format.

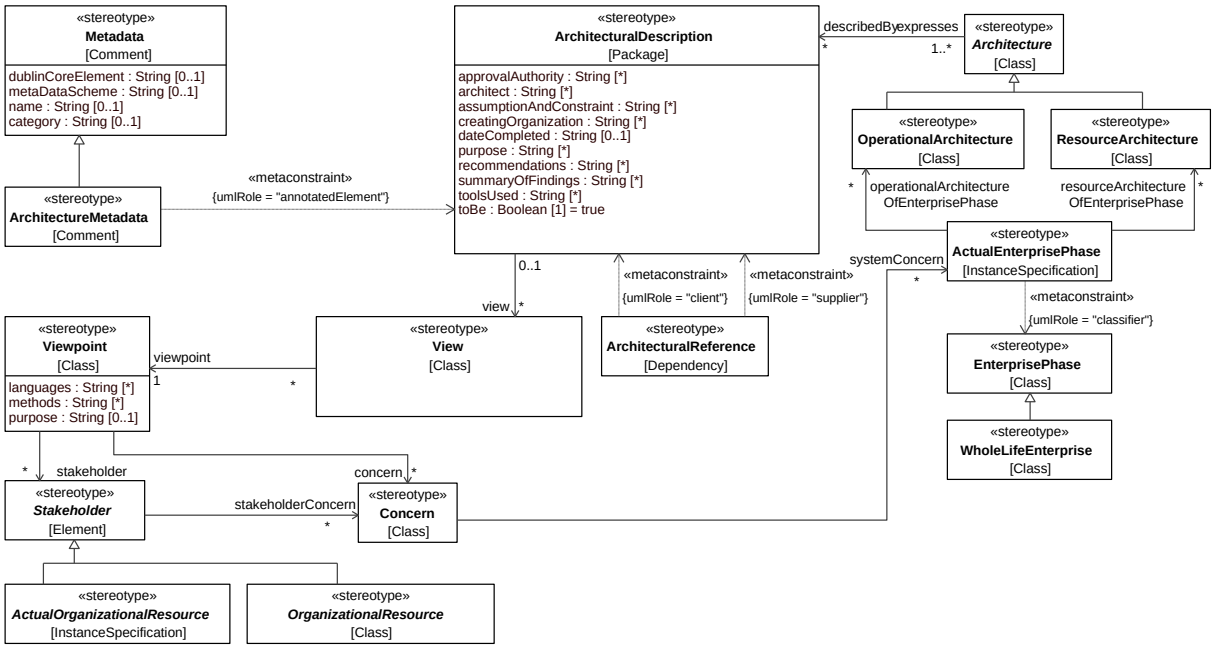

Figure 274 – Summary & Overview

Elements

- ActualEnterprisePhase
- ActualOrganizationalResource
- ArchitecturalDescription
- ArchitecturalReference
- Architecture
- ArchitectureMetadata
- Concern
- EnterprisePhase
- Metadata
- OperationalArchitecture
- OrganizationalResource

- ResourceArchitecture
- Stakeholder
- View
- Viewpoint
- WholeLifeEnterprise

View Specifications::Information

Stakeholders: Data Modelers, Software Engineers, Systems Engineers
Concerns: address the information perspective on operational, service, and resource architectures.
Definition: allows analysis of an architecture's information and data definition aspect, without consideration of implementation specific issues.
Recommended Implementation: SysML Block Definition Diagram.

View Specifications::Information::Information Model

Stakeholders: Data Modelers, Software Engineers, Systems Engineers
Concerns: address the information perspective on operational, service, and resource architectures.
Definition: allows analysis of an architecture's information and data definition aspect, without consideration of implementation specific issues.
Recommended Implementation: SysML Block Definition Diagram.

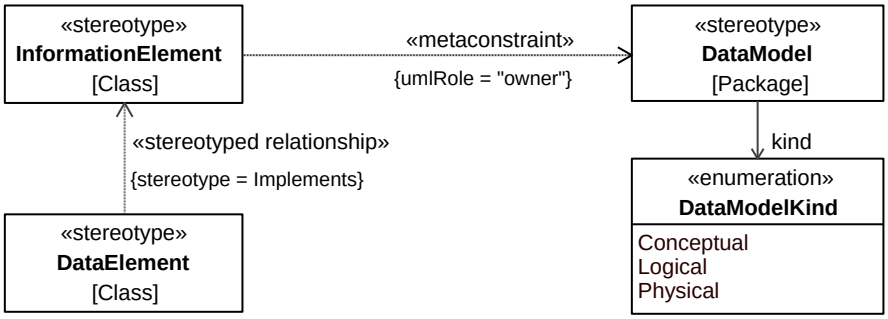

Figure 275 – Information Model

Elements

- DataElement
- DataModel
- DataModelKind
- InformationElement

View Specifications::Parameters

Stakeholders: Capability owners, Systems Engineers, Solution Providers.
Concerns: identifies measurable properties that can be used to support engineering analysis and environment for the Capabilities
Definition: Shows the measurable properties of something in the physical world and elements and relationships that are involved in defining the environments applicable to capability, operational concept or set of systems.

View Specifications::Parameters::Parameters: Environment

Stakeholders: Capability owners, Systems Engineers, Solution Providers.
Concerns: defines the environment for the capabilities.
Definition: shows the elements and relationships that are involved in defining the environments applicable to capability, operational concept or set of systems.
Recommended Implementation: SysML Block Definition Diagram.

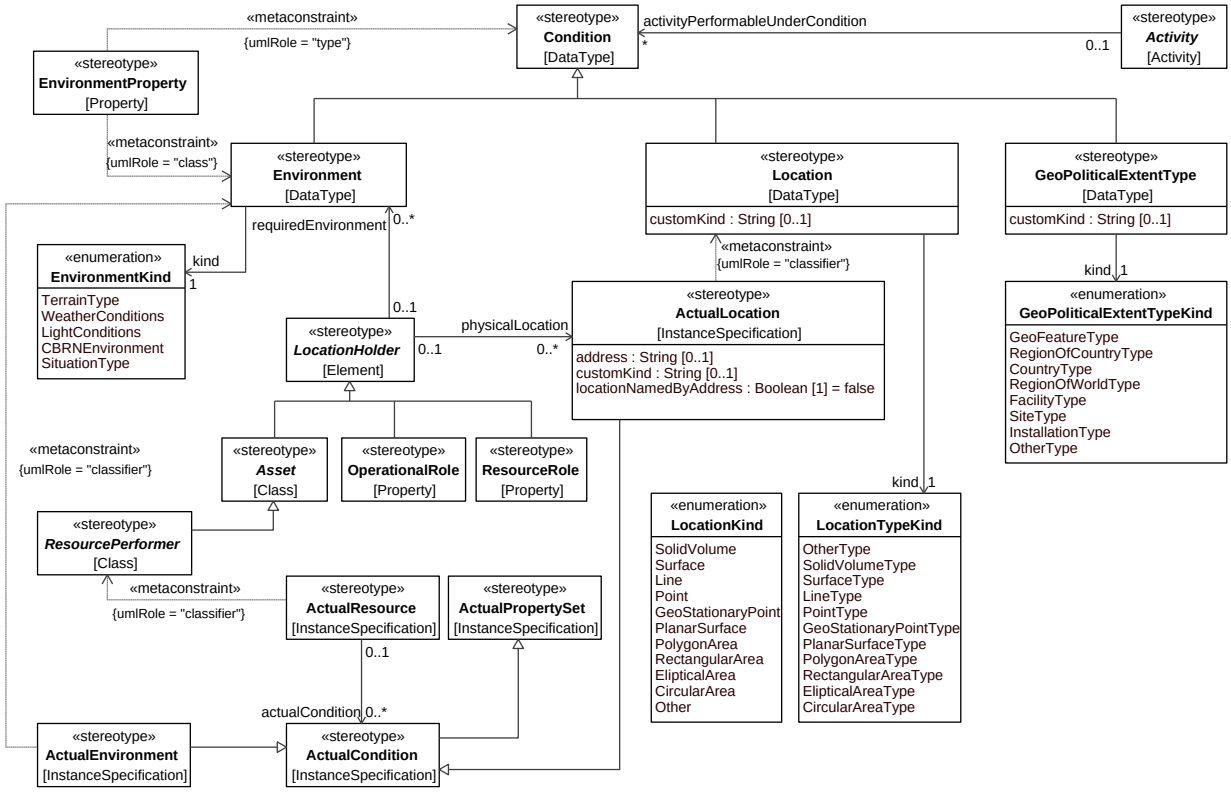

Figure 276 – Parameters: Environment

Elements

- Activity
- ActualCondition
- ActualEnvironment
- ActualLocation
- ActualPropertySet
- ActualResource
- Asset
- Condition
- Environment
- EnvironmentKind
- EnvironmentProperty
- GeoPoliticalExtentType
- GeoPoliticalExtentTypeKind
- Location
- LocationHolder
- LocationKind
- LocationTypeKind
- OperationalRole
- ResourcePerformer
- ResourceRole

View Specifications::Parameters::Parameters: Measurements

Stakeholders: Capability owners, Systems Engineers, Solution Providers.
Concerns: identifies measurable properties that can be used to support analysis such as KPIs, MOs, TPIs etc.
Definition: Shows the measurable properties of something in the physical world, expressed in amounts of a unit of measure that can be associated with any element in the architecture.
Recommended Implementation: SysML Block Definition Diagram.

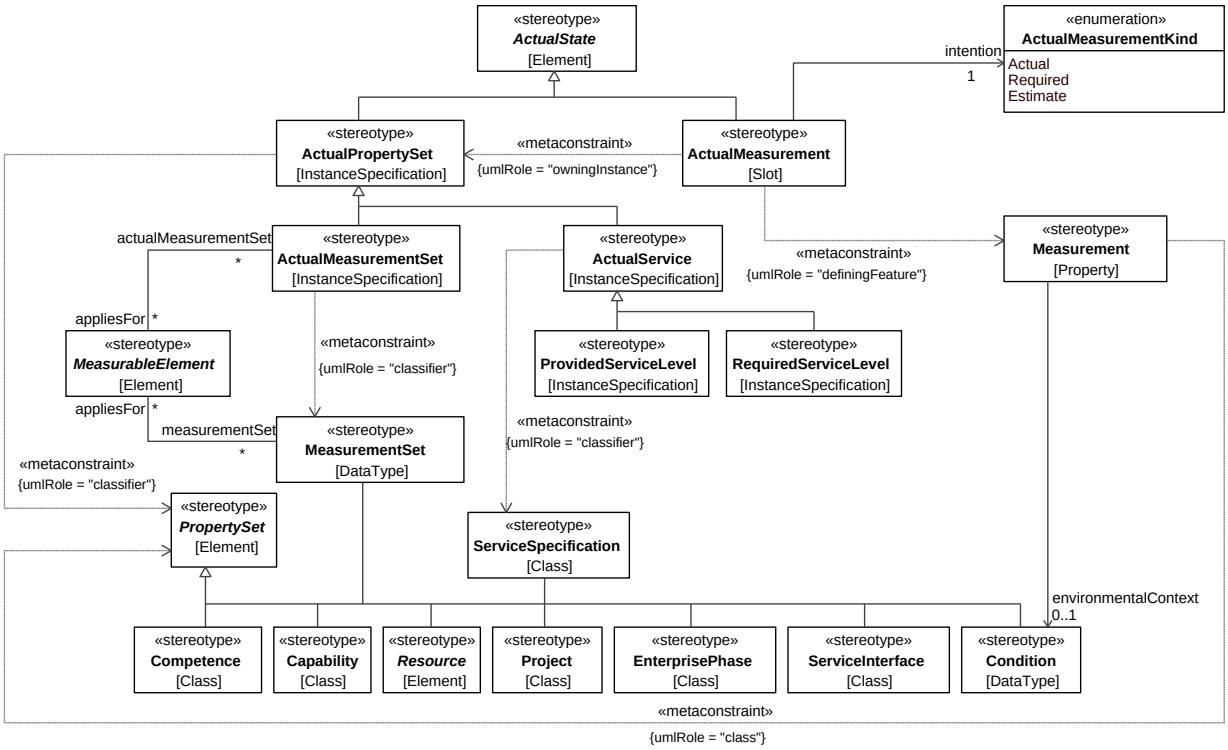

Figure 277 – Parameters: Measurements

Elements

- ActualMeasurement
- ActualMeasurementKind
- ActualMeasurementSet
- ActualPropertySet
- ActualService
- ActualState
- Capability
- Competence
- Condition
- EnterprisePhase
- MeasurableElement
- Measurement
- MeasurementSet
- Project
- PropertySet
- ProvidedServiceLevel
- RequiredServiceLevel
- Resource
- ServiceInterface
- ServiceSpecification

Annex B Class Library

Class Library

A library of Measurements, MeasurementSets and SecurityAttributesGroup, derived from DoDAF.

BillingItem

Package: Class Library

isAbstract: No
Description

Properties indicating the assurance of a piece of information.
Attributes

cost : Cost[1]	Details the cost of the BillingItem.
id : String[0..1]	Details the unique identifier of the BillingItem.
numberOfUses : Integer[0..1]	Details the numberOfUses of the BillingItem.
paymentLocation : String[0..1]	Details the location where payment should be made of the BillingItem.
paymentModality : PricingType[1]	Details if a payment is based upon Quantity, Time or Use.
paymentPeriod : Periodicity[1]	Details the frequency of a payment period.
paymentTimeDuration : Duration[*]	Details the length of time the payments should be made i.e. 1 year.
periodDuration : Duration[0..1]	Details the time period between payments.
quantity : String[0..1]	Details the number of units to be delivered.
unit : String[0..1]	Details the units used for the BillingItem e.g. 1 gross.

ClassificationType

Package: Class Library

isAbstract: No
Description

Enumeration of types of security classification, derived from DoDAF. Its enumeration literals are:

- C - Confidential
- CTS - COSMIC TOP SECRET
- CTS-B - COSMIC TOP SECRET - BOHEMIA
- CTS-BALK - COSMIC TOP SECRET - BALK
- CTSA - COSMIC TOP SECRET ATOMAL
- NC - NATO Confidential
- NCA - NATO Confidential Atomal
- NR - NATO Restricted (similar to US For Official Use only)
- NS - NATO Secret
- NS-A - NATO Atomal
- NS-S - NATO Secret
- NSAT - NATO Secret Atomal
- NU - NATO Unclassified
- R - Restricted Data (RD) US Nuclear Information OR FOR OFFICIAL USE ONLY
- S - Secret
- TS - Top Secret
- U - Unclassified

CommunicationsLinkProperties

Package: Class Library

isAbstract: No
Description

Properties detailing aspects of Resource Interfaces.
Attributes

capacity : String[]	Details how much information can be passed on the Communications Link.
infrastructureTechnology : String[]	Details the technology to be used to provide the communications infrastructure.

DataElementProperties

Package: Class Library

isAbstract: No
Description

Properties detailing the aspects of a DataElement.
Attributes

accuracy : String[]	Details the accuracy of the data.
content : String[]	Specifies content of the data element (i.e., actual data to be exchanged).
formatType : String[]	Details the format of the data.
mediaType : String[]	Details the media used to transmit the data.
scope : String[]	Details in text a description of the extent or range of the data element content.
unitOfMeasurement : String[]	Details the units of measurement of the data.

Duration

Package: Class Library

isAbstract: No
Description

Properties detailing aspects OperationalActivities.
Attributes

timeUnit : String[0..1]	Details the units of time e.g. second, hour, day.
value : Integer[0..1]	Details the value of the duration.

ExchangeProperties

Package: Class Library

isAbstract: No
Description

Properties detailing aspects of exchange for Operational Exchange and/or Resource Interaction.
Attributes

accountability : String[*]	Details who or what is responsible for the exchange.
classificationCaveat : String[*]	Details any caveats to the classification.
criticality : String[*]	Details the criticality assessment of the information being exchanged in relationship to the mission being performed.
interoperabilityLevelAchievable : String[*]	Details the Level of Information Systems Interoperability (LISI) achieved or achievable through the exchange.
periodicity : String[*]	Details the frequency of the exchange.
protectionDuration : String[*]	Details the duration of protection of the exchange.
protectionDurationCode : String[*]	Details the code that represents how long the exchange must be

	safeguarded.
protectionSuspenseCalendarDate : String[*]	Details the date when protection for the exchange is suspended.
protectionTypeName : String[*]	Details the name for the type of the protection on the exchange.
releasability : String[*]	Details the code that represents the kind of controls required for further dissemination of information from the exchange.
size : String[*]	Details the size (in KB) of data that be exchanged.
Taxonomy : String[*]	Details a classification for the exchange.
throughput : String[*]	Details how much information can be exchanged.
timeliness : String[*]	Details the allowable time of delay this system data can tolerate and still be relevant to the receiving system.
transactionType : String[*]	Details the type of transactions used by the exchange.

InformationAssuranceProperties

Package: Class Library

isAbstract: No

Description

Properties indicating the assurance of a piece of information.

Attributes

accessControl : String[*]	Details the class of mechanisms used to ensure that only those authorized get access to specific systems and information.
availability : String[*]	Details how timely and reliable access to and use of information is achieved.
confidentiality : String[*]	Details the kind of authorized restrictions on information access and disclosure, including means for protecting personal privacy and proprietary information.
disseminationControl : String[*]	Details the kind of restrictions on receivers of data based on the sensitivity of data.
integrity : String[*]	Details the kind of restrictions used for guarding against improper information modification or destruction, and includes ensuring information non-repudiation and authenticity.
nonRepudiationConsumer : String[*]	Details the requirements for unassailable knowledge that the system data sent was consumed by the intended recipient.
nonRepudiationProducer : String[*]	Details the requirements for unassailable knowledge that the system data received was produced by the stated source.

InformationElementProperties

Package: Class Library

isAbstract: No

Description

Predefined additional DoDAF properties for InformationElement.

Attributes

accuracy : String[*]	Details the degree to which the information conforms to actual fact as required by the information producer and consumer.
content : String[*]	Specifies content of the information element (i.e., actual information to be exchanged).
language : String[*]	Details the language used to capture the information.
scope : String[*]	Details in text a description of the extent or range of the information element content.

OperationalActivityProperties

Package: Class Library

isAbstract: No
Description

Properties detailing aspects OperationalActivities.
Attributes

cost : String[] Details the cost of an activity.

Periodicity

Package: Class Library

isAbstract: No
Description

Enumeration of how often the information exchange occurs; may be an average or a worst case estimate and may include conditions. Its enumeration literals are:

- OnceAMonth - Indicates that an event of some sort may occur monthly.
- OnceAWeek - Indicates that an event of some sort may occur weekly.
- Anytime - Indicates that an event of some sort may occur at anytime.
- OnRequest - Indicates that an event of some sort may occur on request.

PricingType

Package: Class Library

isAbstract: No
Description

Enumeration of a unit of measure of a resource. Its enumeration literals are:

- perTIme - Indicates that the unit of measure of a resource is based on a unit of time.
- perUse - Indicates that the unit of measure of a resource is based upon how often the resource is used.
- perQuantity - Indicates that the unit of measure of a resource is based on a quantity.

SecurityAttributes

Package: Class Library

isAbstract: No
Description

W3C XML Schema for the Intelligence Community Metadata Standard for Information Security Marking (IC-ISM), which is part of the IC standards for Information Assurance.
Attributes

classificationReason : String[]	One or more reason indicators or explanatory text describing the basis for an original classification decision.
classifiedBy : String[]	Details The identity, by name or personal identifier, and position title of the original classification authority for a resource.
dateOfExemptedSource : String[]	Details the specific year, month, and day of publication or release of a source document, or the most recent source document, that was itself marked with a declassification constraint. This element is always used in conjunction with typeOfExemptedSource element.
declassDate : String[]	Details a specific year, month, and day upon which the information shall be automatically declassified if not properly exempted from automatic declassification.
declassEvent : String[]	Details a description of an event upon which the information shall be automatically declassified if not properly exempted from automatic declassification.
declassException : String[]	Details a single indicator describing an exemption to the nominal 25-year point for automatic declassification. This element is used in conjunction with the Declassification Date or Declassification Event.
DeclassManualReview : String[]	Details a true/false indicator that a manual review is required for declassification.

	Use this attribute to force the appearance of "//MR" in the header and footer marking titles. Use this attribute ONLY when it is necessary to override the business logic applied to classification and control markings in the document to determine whether manual review is required.
derivedFrom : String[]	Details a citation of the authoritative source or reference to multiple sources of the classification markings used in a classified resource.
DisseminationControls : String[]	Details one or more indicators identifying the expansion or limitation on the distribution of information.
FGIsourceOpen : String[]	Details one or more indicators identifying information which qualifies as foreign government information for which the source(s) of the information is not concealed.
FGIsourceProtected : String[]	Details a single indicator that information qualifies as foreign government information for which the source(s) of the information must be concealed. Within protected internal organizational spaces this element may be used to maintain a record of the one or more indicators identifying information which qualifies as foreign government information for which the source(s) of the information must be concealed. Measures must be taken prior to dissemination of the information to conceal the source(s) of the foreign government information.
nonICmarkings : String[]	Details one or more indicators of the expansion or limitation on the distribution of an information resource or portion within the domain of information originating from non-intelligence components.
ownerProducer : String[]	Details one or more indicators identifying the national government or international organization that have purview over the classification marking of an information resource or portion therein. This element is always used in conjunction with the Classification element. Taken together, the two elements specify the classification category and the type of classification (US, non-US, or Joint). Within protected internal organizational spaces this element may include one or more indicators identifying information which qualifies as foreign government information for which the source(s) of the information must be concealed. Measures must be taken prior to dissemination of the information to conceal the source(s) of the foreign government information.
releasableTo : String[]	Details one or more indicators identifying the country or countries and/or international organization(s) to which classified information may be released based on the determination of an originator in accordance with established foreign disclosure procedures. This element is used in conjunction with the Dissemination Controls element.
SARIdentifier : String[]	Details the Authorized Special Access Required (SAR) program digraph(s) or trigraph(s) preceded by "SAR-". Either (a) a single digraph or trigraph or (b) a space-delimited list of digraphs or trigraphs. Example: "SAR-ABC SAR-DEF ..."
SCIControls : String[]	Details one or more indicators identifying sensitive compartmented information control system(s).
typeOfExemptedSource : String[]	Details a declassification marking of a source document that causes the current, derivative document to be exempted from automatic declassification. This element is always used in conjunction with the Date Of Exempted Source element.
Associations	
taxonomy : ClassificationType[]	Details a single indicator of the highest level of classification applicable to an information resource or portion within the domain of classified national security information. The Classification element is always used in conjunction with the Owner Producer element. Taken together, the two elements specify the classification category and the type of classification (US, non-US, or Joint).

SecurityCategoryProperties

Package: Class Library

isAbstract: No
Description

Properties detailing aspects of Security Categories.
Attributes

criticality : String[1]	Details the criticality assessment of the information being exchanged in relationship to the mission being performed.
maturityOfControl : String[1]	Details the level of control (High, Medium, Low).
securityAvailability : String[*]	Details how the security category is made available for those who need access to it.
securityClassification : String[*]	Details a classification for the exchange.
securityConfidentiality : String[*]	Details the classification type of the security.
securityIntegrity : String[*]	Details the kind of restrictions used for guarding against improper information modification or destruction, and includes ensuring information non-repudiation and authenticity.

SecurityControlProperties

Package: Class Library

isAbstract: No
Description

Properties detailing aspects of Security Controls.
Attributes

criticality : String[1]	Details how important and critical the security control is.
maturityOfControl : String[1]	Details how mature the security control is.