USE OF SPECIFICATION - TERMS, CONDITIONS & NOTICES

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

LICENSES

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

Subject to all of the terms and conditions below, the owners of the copyright in this specification hereby grant you a fully-paid up, non-exclusive, nontransferable, perpetual, worldwide license (without the right to sublicense), to use this specification to create and distribute software and special purpose specifications that are based upon this specification, and to use, copy, and distribute this specification as provided under the Copyright Act; provided that:

1. both the copyright notice identified above and this permission notice appear on any copies of this specification;
2. the use of the specifications is for informational purposes and will not be copied or posted on any network computer or broadcast in any media and will not be otherwise resold or transferred for commercial purposes; and
3. no modifications are made to this specification. This limited permission automatically terminates without notice if you breach any of these terms or conditions. Upon termination, you will destroy immediately any copies of the specifications in your possession or control.

PATENTS

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

GENERAL USE RESTRICTIONS

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

WHILE THIS PUBLICATION IS BELIEVED TO BE ACCURATE, IT IS PROVIDED "AS IS" AND MAY
CONTAIN ERRORS OR MISPRINTS. THE OBJECT MANAGEMENT GROUP AND THE COMPANIES
LISTED ABOVE MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS
PUBLICATION, INCLUDING BUT NOT LIMITED TO ANY WARRANTY OF TITLE OR OWNERSHIP,
IMPLIED WARRANTY OF MERCHANTABILITY OR WARRANTY OF FITNESS FOR A PARTICULAR
PURPOSE OR USE. IN NO EVENT SHALL THE OBJECT MANAGEMENT GROUP OR ANY OF THE
COMPANIES LISTED ABOVE BE LIABLE FOR ERRORS CONTAINED HEREIN OR FOR DIRECT,
INDIRECT, INCIDENTAL, SPECIAL, CONSEQUENTIAL, RELIANCE OR COVER DAMAGES, INCLUDING
LOSS OF PROFITS, REVENUE, DATA OR USE, INCURRED BY ANY USER OR ANY THIRD PARTY IN
CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS MATERIAL, EVEN IF
ADvised OF THE POSSIBILITY OF SUCH DAMAGES.

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

RESTRICTED RIGHTS
LEGEND

Use, duplication or disclosure by the U.S. Government is subject to the restrictions set forth in subparagraph (c) (1)
(ii) of The Rights in Technical Data and Computer Software Clause at DFARS 252.227-7013 or in subparagraph (c)
(1) and (2) of the Commercial Computer Software - Restricted Rights clauses at 48 C.F.R. 52.227-19 or as specified in
Federal Acquisition Regulations and its successors, as applicable. The specification copyright owners are as
indicated above and may be contacted through the Object Management Group, 109 Highland Avenue, Needham,
MA 02494, U.S.A.

TRADEMARK
S

CORBA®, CORBA logos®, FIBO®, Financial Industry Business Ontology®, FINANCIAL INSTRUMENT
GLOBAL IDENTIFIER®, IIOP®, IMM®, Model Driven Architecture®, MDA®, Object Management Group®,
OMG®, OMG Logo®, SoaML®, SOAML®, SysML®, UAF®, Unified Modeling Language®, UML®, UML Cube
Logo®, VSIPL®, and XMI® are registered trademarks of the Object Management Group, Inc.
For a complete list of trademarks, see: http://www.omg.org/legal/tm_list.htm. All other products or company names
mentioned are used for identification purposes only, and may be trademarks of their respective owners.

COMPLIANCE
E

The copyright holders listed above acknowledge that the Object Management Group (acting itself or through its
designees) is and shall at all times be the sole entity that may authorize developers, suppliers and sellers of computer
software to use certification marks, trademarks or other special designations to indicate compliance with these
materials.
Software developed under the terms of this license may claim compliance or conformance with this specification if
and only if the software compliance is of a nature fully matching the applicable compliance points as stated in the
specification. Software developed only partially matching the applicable compliance points may claim only that the
software was based on this specification, but may not claim compliance or conformance with this specification. In
the event that testing suites are implemented or approved by Object Management Group, Inc., software developed
using this specification may claim compliance or conformance with the specification only if the software
satisfactorily completes the testing suites.
OMG's Issue Reporting Procedure

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

PREFACE .................................................................................................................................................. 15

1. SCOPE .................................................................................................................................................. 15
   1.1 INTRODUCTION ................................................................................................................................. 15
   1.2 UAF BACKGROUND ............................................................................................................................. 15
   1.3 INTENDED USAGE ............................................................................................................................... 16
   1.4 RELATED DOCUMENTS ....................................................................................................................... 16

2. CONFORMANCE ..................................................................................................................................... 17

3. REFERENCES .......................................................................................................................................... 18
   3.1 NORMATIVE REFERENCES .................................................................................................................. 18
   3.2 OMG DOCUMENTS (NORMATIVE REFERENCES) ............................................................................... 18
   3.3 OTHER NORMATIVE REFERENCES ................................................................................................... 18
   3.4 INFORMATIVE REFERENCES ............................................................................................................. 18

4. TERMS AND DEFINITIONS ....................................................................................................................... 20

5. SYMBOLS ............................................................................................................................................... 21

6. ADDITIONAL INFORMATION .................................................................................................................. 23
   6.1 CHANGES TO ADOPTED OMG SPECIFICATIONS ............................................................................. 23
   6.2 LANGUAGE ARCHITECTURE ................................................................................................................ 23
   6.3 PHILOSOPHY ....................................................................................................................................... 23
   6.4 CORE PRINCIPLES ............................................................................................................................... 23

7. UAF GRID ............................................................................................................................................... 24
   7.1 DESCRIPTIONS OF DOMAINS AND MODEL KINDS .......................................................................... 26
   7.2 DOMAIN INTERRELATIONSHIPS ........................................................................................................ 28

8. DOMAIN METAMODEL DIAGRAM LEGEND ......................................................................................... 28

9. DOMAIN METAMODEL DIAGRAMS ....................................................................................................... 30
   9.1 VIEW SPECIFICATIONS ....................................................................................................................... 30
      9.1.1 View Specifications::Metadata ..................................................................................................... 30
      9.1.2 View Specifications::Strategic ..................................................................................................... 31
      9.1.3 View Specifications::Operational ................................................................................................ 31
      9.1.4 View Specifications::Services ....................................................................................................... 31
      9.1.5 View Specifications::Personnel .................................................................................................... 31
      9.1.6 View Specifications::Resources .................................................................................................... 31
      9.1.7 View Specifications::Security ....................................................................................................... 31
      9.1.8 View Specifications::Projects ...................................................................................................... 31
      9.1.9 View Specifications::Other ........................................................................................................... 31
      9.1.10 View Specifications::Actual Resources ...................................................................................... 31
      9.1.11 View Specifications::Dictionary .................................................................................................. 31
      9.1.12 View Specifications::Summary & Overview ............................................................................. 31
      9.1.13 View Specifications::Requirements .......................................................................................... 31
      9.1.14 View Specifications::Information ............................................................................................... 31
      9.1.15 View Specifications::Parameters ................................................................................................ 31
      9.1.16 View Specifications::Other ........................................................................................................ 31

10. DOMAIN METAMODEL (DMM) ELEMENTS .......................................................................................... 30
    10.1 DOMAIN METAMODEL .................................................................................................................... 30
       10.1.1 Domain MetaModel::Metadata .................................................................................................... 30
       10.1.2 Domain MetaModel::Strategic .................................................................................................... 31
       10.1.3 Domain MetaModel::Operational ............................................................................................... 31
       10.1.4 Domain MetaModel::Services .................................................................................................... 31
       10.1.5 Domain MetaModel::Requirements ........................................................................................... 31
       10.1.6 Domain MetaModel::Information ............................................................................................... 31
       10.1.7 Domain MetaModel::Parameters ............................................................................................... 31
       10.1.8 Domain MetaModel::Other ....................................................................................................... 31

Unified Architecture Framework (UAF), v1.0
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.3.4.1 View Specifications::Services::Processes: Services Processes</td>
<td>26</td>
</tr>
<tr>
<td>2.3.5 View Specifications::Services::States: States</td>
<td>27</td>
</tr>
<tr>
<td>2.3.6 View Specifications::Services::Interaction Scenarios</td>
<td>28</td>
</tr>
<tr>
<td>2.3.7 View Specifications::Services::Constraints: Constraints</td>
<td>28</td>
</tr>
</tbody>
</table>
## Table of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1.1</td>
<td>UAF Grid Overview</td>
<td>2</td>
</tr>
<tr>
<td>Figure 1.2</td>
<td>Legend of color codes for element types defined in UAF</td>
<td>5</td>
</tr>
<tr>
<td>Figure 2.1</td>
<td>Strategic Taxonomy</td>
<td>6</td>
</tr>
<tr>
<td>Figure 2.2</td>
<td>Strategic Structure</td>
<td>7</td>
</tr>
<tr>
<td>Figure 2.3</td>
<td>Strategic Connectivity</td>
<td>8</td>
</tr>
<tr>
<td>Figure 2.4</td>
<td>Strategic States</td>
<td>9</td>
</tr>
<tr>
<td>Figure 2.5</td>
<td>Strategic Constraints</td>
<td>10</td>
</tr>
<tr>
<td>Figure 2.6</td>
<td>Strategic Roadmap: Deployment</td>
<td>11</td>
</tr>
<tr>
<td>Figure 2.7</td>
<td>Strategic Roadmap: Phasing</td>
<td>12</td>
</tr>
<tr>
<td>Figure 2.8</td>
<td>Strategic Traceability</td>
<td>13</td>
</tr>
<tr>
<td>Figure 2.9</td>
<td>Operational Taxonomy</td>
<td>14</td>
</tr>
<tr>
<td>Figure 2.10</td>
<td>Operational Structure</td>
<td>16</td>
</tr>
<tr>
<td>Figure 2.11</td>
<td>Operational Connectivity</td>
<td>17</td>
</tr>
<tr>
<td>Figure 2.12</td>
<td>Operational Processes</td>
<td>19</td>
</tr>
<tr>
<td>Figure 2.13</td>
<td>Operational States</td>
<td>20</td>
</tr>
<tr>
<td>Figure 2.14</td>
<td>Operational Interaction Scenarios</td>
<td>21</td>
</tr>
<tr>
<td>Figure 2.15</td>
<td>Operational Constraints</td>
<td>22</td>
</tr>
<tr>
<td>Figure 2.16</td>
<td>Operational Traceability</td>
<td>23</td>
</tr>
<tr>
<td>Figure 2.17</td>
<td>Services Taxonomy</td>
<td>24</td>
</tr>
<tr>
<td>Figure 2.18</td>
<td>Services Structure</td>
<td>25</td>
</tr>
<tr>
<td>Figure 2.19</td>
<td>Services Connectivity</td>
<td>26</td>
</tr>
<tr>
<td>Figure 2.20</td>
<td>Services Processes</td>
<td>27</td>
</tr>
<tr>
<td>Figure 2.21</td>
<td>Services States</td>
<td>27</td>
</tr>
<tr>
<td>Figure 2.22</td>
<td>Services Interaction Scenarios</td>
<td>28</td>
</tr>
<tr>
<td>Figure 2.23</td>
<td>Services Constraints</td>
<td>29</td>
</tr>
<tr>
<td>Figure 2.24</td>
<td>Services Roadmap</td>
<td>30</td>
</tr>
<tr>
<td>Figure 2.25</td>
<td>Services Traceability</td>
<td>31</td>
</tr>
<tr>
<td>Figure 2.26</td>
<td>Personnel Taxonomy</td>
<td>32</td>
</tr>
<tr>
<td>Figure 2.27</td>
<td>Personnel Structure</td>
<td>33</td>
</tr>
<tr>
<td>Figure 2.28</td>
<td>Personnel Connectivity</td>
<td>34</td>
</tr>
<tr>
<td>Figure 2.29</td>
<td>Personnel Processes</td>
<td>36</td>
</tr>
<tr>
<td>Figure 2.30</td>
<td>Personnel States</td>
<td>37</td>
</tr>
<tr>
<td>Figure 2.31</td>
<td>Personnel Interaction Scenarios</td>
<td>38</td>
</tr>
<tr>
<td>Figure 2.32</td>
<td>Personnel Constraints: Competence</td>
<td>39</td>
</tr>
<tr>
<td>Figure 2.33</td>
<td>Personnel Constraints: Drivers</td>
<td>40</td>
</tr>
<tr>
<td>Figure 2.34</td>
<td>Personnel Constraints: Performance</td>
<td>42</td>
</tr>
<tr>
<td>Figure 2.35</td>
<td>Personnel Roadmap: Availability</td>
<td>44</td>
</tr>
<tr>
<td>Figure 2.36</td>
<td>Personnel Roadmap: Evolution</td>
<td>45</td>
</tr>
<tr>
<td>Figure 2.37</td>
<td>Personnel Roadmap: Forecast</td>
<td>46</td>
</tr>
<tr>
<td>Figure 2.38</td>
<td>Personnel Traceability</td>
<td>47</td>
</tr>
<tr>
<td>Figure 2.39</td>
<td>Resources Taxonomy</td>
<td>48</td>
</tr>
<tr>
<td>Figure 2.40</td>
<td>Resources Structure</td>
<td>49</td>
</tr>
<tr>
<td>Figure 2.41</td>
<td>Resources Connectivity</td>
<td>51</td>
</tr>
<tr>
<td>Figure 2.42</td>
<td>Resources Processes</td>
<td>53</td>
</tr>
<tr>
<td>Figure 2.43</td>
<td>Resources States</td>
<td>54</td>
</tr>
<tr>
<td>Figure 2.44</td>
<td>Resources Interaction Scenarios</td>
<td>54</td>
</tr>
<tr>
<td>Figure 2.45</td>
<td>Resources Constraints</td>
<td>55</td>
</tr>
<tr>
<td>Figure 2.46</td>
<td>Resources Roadmap: Evolution</td>
<td>57</td>
</tr>
<tr>
<td>Figure 2.47</td>
<td>Resources Roadmap: Forecast</td>
<td>58</td>
</tr>
<tr>
<td>Figure 2.48</td>
<td>Resources Traceability</td>
<td>59</td>
</tr>
<tr>
<td>Figure 2.49</td>
<td>Security Taxonomy</td>
<td>60</td>
</tr>
<tr>
<td>Figure 2.50</td>
<td>Security Structure</td>
<td>61</td>
</tr>
</tbody>
</table>
This page intentionally left blank.
1 Scope

1.1 Introduction

There are four parts to this specification, two are normative and two informative. The normative parts are:

1. The UAF Domain Metamodel (DMM) (this document) that provides the definition of concepts, relationships and viewpoints for the framework. The UAF DMM is the basis for any implementation of UAF including non-UML/SysML implementations.

2. The UAF Profile (UAFP) is a UML/SysML implementation of the UAF DMM.

The informative parts are:

3. The UAF Traceability, Annex A, which details the mappings between the UAF and the various frameworks and languages that contribute to the UAF.

4. The UAF Example Model, Annex B, which illustrates a practical usage of UAF.

1.2 UAF Background

UAF evolved from the Unified Profile for DoDAF and MODAF (UPDM), version 2.1. UAF extends the scope of UPDM and generalizes it to make it applicable to commercial as well as military architectures. The intent of UAF is to provide a standard representation for describing enterprise architectures using a Model Based Systems Engineering (MBSE) approach.

The core concepts in the UAF are based upon the DoDAF 2.0.2 Domain Metamodel (DM2) and the MODAF ontological data exchange mechanism (MODEM), Security Views from Canada's Department of National Defense Architecture Framework (DNDAF) and the North Atlantic Treaty Organization (NATO) Architecture Framework (NAF) v 4.

UAF models describe a system from a set of stakeholders’ concerns such as security or information through a set of predefined viewpoints. Developed models can also reflect custom viewpoints or users can develop more formal extensions for new viewpoints.

The UAFP can be used to develop architectures compliant with:

- Department of Defense Architecture Framework (DoDAF) version 2.02
- Ministry of Defence Architecture Framework (MODAF) version 1.3
- North Atlantic Treaty Organization (NATO) Architecture Framework (NAF) version 3.1
- North Atlantic Treaty Organization (NATO) Architecture Framework (NAF) version 4

UAF v 1.1 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.

---

1 The term system is used from: “Systems and software engineering -- Architecture description,” http://www.iso.org/iso/catalogue_detail.htm?csnumber=50508
1.3 Intended Usage

The UAF enables the modeling of strategic capabilities, operational scenarios, services, resources, personnel, security, projects, standards, measures and requirements; which supports best practices through, separation of concerns and abstractions. In addition, the UAF 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)
- UK Ministry of Defence Lines of Development (DLOD) elements,
- Human Computer Interfaces (HCI).

Further, The UAF 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 UAF.

1.4 Related Documents

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

<table>
<thead>
<tr>
<th>Table 1:1 - Table of Related Documents</th>
</tr>
</thead>
<tbody>
<tr>
<td>dtc/19-06-1616-05-01</td>
</tr>
<tr>
<td>dtc/19-06-1516-05-02</td>
</tr>
<tr>
<td>dtc/19-06-176-05-03</td>
</tr>
<tr>
<td>dtc/19-06-186-05-04</td>
</tr>
<tr>
<td>dtc/19-06-196-05-05</td>
</tr>
<tr>
<td>dtc/19-06-2016-05-06</td>
</tr>
<tr>
<td>dtc/19-05-146-05-02</td>
</tr>
</tbody>
</table>
2. Conformance

UAF specifies four types of conformance.

Type 1 Conformance: UAF View specification conformance. A tool demonstrating view specification conformance shall implement a version of all the view specifications defined in the UAF Grid, with the exception of the view specifications in the Metadata Domain. Optionally the tool vendor can implement other donor framework viewpoints, for instance DoDAF, MODAF or NAF based upon the mapping between them and UAF provided in UAF traceability document XXXX Appendix A (dtc/19-06-17).

Type 2 Conformance: UAF Conceptual Syntax Conformance. A tool demonstrating conceptual syntax conformance is consistent with the concepts, relationships, and constraints defined in the UAF DMM (this document). UAF Conceptual Syntax Conformance implies Type 1 Conformance.

Type 3 Conformance: UAF Formal Syntax Conformance. A tool demonstrating formal syntax conformance:

- enables instances of concrete UAFP stereotypes defined in the UAFP (dtc/19-06-15 document xxxx)
- complies with the constraints defined in the UAFP (dtc/19-06-15 document xxxx)
- complies with the SysML version 1.5 Concrete Syntax Conformance (formal/17-05-01) (document xxxx)

UAF Formal Syntax Conformance implies Type 2 Conformance.

Type 4 Conformance: UAF Model interchange conformance. A tool demonstrating model interchange conformance can import and export conformant XMI for all valid UAFP models. Model interchange conformance implies Type 3 Conformance.
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 Profile for DoDAF and MODAF (UPDM), 2.1, August 2013, http://www.omg.org/spec/UPDM

3.3 Other Normative References

- DM2 - DoDAF Meta-Model
- IDEAS Foundation v1.0 as XMI File (zipped), http://www.ideasgroup.org/7Documents/
- MODAF Ontological Data Exchange Mechanism (MODEM)
- NATO Architecture Framework (NAF),
- NATO Architecture Framework v4.0 Documentation.

3.4 Informative References

• ISO 8601:2004 Data elements and interchange formats – Information interchange – Representation of dates and times,
• 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
  (Authoritative)
• Object Management Group (OMG), Semantics of Business Vocabulary and Business Rules (SBVR), Version 1.3, May
• 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-*² | 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 |
| STV-* | Strategic View |
| SV-* | System View |

² * denotes a wildcard

Unified Architecture Framework (UAF), v1.0
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SvcV.*</td>
<td>Service View</td>
</tr>
<tr>
<td>TEPID OIL</td>
<td>Training, Equipment, Personnel, Information, Concepts and Doctrine, Organisation, Infrastructure, Logistics</td>
</tr>
<tr>
<td>TOGAF</td>
<td>The Open Group Architectural Framework©</td>
</tr>
<tr>
<td>TPPU</td>
<td>Task, Post, Process, and Use</td>
</tr>
<tr>
<td>TV.*</td>
<td>Technical View</td>
</tr>
<tr>
<td>UAF</td>
<td>Unified Architecture Framework</td>
</tr>
<tr>
<td>UAFP</td>
<td>Unified Architecture Framework Profile</td>
</tr>
<tr>
<td>UPDM</td>
<td>Unified Profile for DoDAF/MODAF</td>
</tr>
</tbody>
</table>
6. Additional Information

6.1 Changes to Adopted OMG Specifications

This specification completely replaces Unified Architecture Framework (UAF), version 1.0. https://www.omg.org/spec/UAF/About-UAF/

6.2 Language Architecture

The UAF specification reuses a subset of UML 2.5.1 and SysML 1.5 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 UML 2.5.1 and SysML 1.5 and specifies how to implement UAF. This clause explains design principles and how they are applied to define the UAF language architecture.

6.3 Philosophy

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

- A Domain Metamodel (DMM) uses UML Class models to represent individuals, types and tuples that aggregate the concepts defined in DoDAF, MODEM, NAF, DNDAF and other frameworks.

- 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 UAF 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 metamodel from the presentation layer. The results of this mapping are given in Appendix AB (see document dtc/19-06-176-05-03), and an overview of the viewpoints in a grid format are given in this document.

- The intent of the UAF is to provide a Domain MetaModel usable by non-UML/SysML tool vendors who may wish to implement the UAF within their own tool and metalanguage.

- The Unified Architecture Framework Profile (UAFP) is the standard implementation of the UAF DMM. It was created by mapping the UAF concepts and relationships to corresponding stereotypes in the UAFP.

- The UAFP analysis and refactoring reflects language architecture, tool implementation, and reuse considerations.

- The specification is generated from the UML model used to describe the UAF DMM and UAFP. This approach allows the team to concentrate on architecture issues rather than documentation production. The UML tool automatically maintains consistency. The UML tool improves maintenance and enables traceability between the UAF and the UAFP where every stereotype is linkable to the UAF element using UML Abstraction relationship.

6.4 Core Principles

The fundamental design principles for UAF DMM are:

- **Requirements-driven**: UAF is intended to satisfy the requirements of the UPDM 3.0 RFP Mandatory Requirements.

- **Influence from donor Frameworks**: The DMM was based upon an aggregation of concepts and relationships from the donor frameworks.

- **IDEAS Ontology driven**: The DMM was based upon a simplified version of the IDEAS ontology, see chapter 8.

- **DMM Notation**: The DMM was expressed using UML class diagram notation.

- **Reusability of UML Metamodel concepts**: The UAF DMM reuses a number of concepts from the UML Metamodel, such as Statemachines, Activities and Interactions. The explicit relationship to these concepts enables the UAF DMM to reuse UML semantics instead of reinventing its own semantics.

- **Reusability of BPMN concepts**: The UAF DMM reuses a number of concepts from BPMN, such as processes. The explicit relationship to these concepts enables the UAF DMM to reuse BPMN semantics instead of reinventing its own semantics.
7.4 UAF Grid

Due to the complexity of managing the multiple viewpoints with overlapping concerns and metamodels, the standard viewpoints are refactored as described in the donor frameworks into a more manageable format. This decision led to the development of the UAF grid which is described below.

The grid is a way of showing how the various viewpoints (known as view specifications in the rest of document) correspond to domains (horizontal rows) and the model kinds (the columns) that describe the view specification. The intent of the grid is not to be complete, but to capture the information that is present in the frameworks that contributes to the UAF, consequently, some gaps are evident.

![UAF Grid Table]

Figure 7:1 - UAF Grid

Notes related to suffixes in the grid

a. The view specifications in the Metadata Domain are not modeled as part of the UAF but are architectural artifacts that contribute to the success in defining and developing an architecture.

b. To be able to evaluate architecture behavior and constraints (i.e., non-functional requirements) it is necessary to define actual instances of the architectural elements. The expectation is that tool vendors intending to implement the UAF have capabilities native to their tools to enable behavioral simulation and the evaluation of measures and constraints through parametric diagrams or a proprietary equivalent.

c. The information model is a column across the domains and can be defined in any of its forms, i.e., Conceptual, Logical or Physical. The expectation is that most developers of the information model will use the Conceptual or Logical forms of the data model when using an abstract modeling tool.

d. The parameters column captures the measures and environments across the architecture in all the different domains.
e. The expectation is that the physical schema model would not be defined in the UAF. Any tool implementing the framework provides a means to import or link-to representations of the physical model.

f. The Metadata Taxonomy view specification provides a means to extend the framework to other domains.

The detailed mapping between the view specifications of the UAF shown in the grid and the viewpoints from the donor frameworks is described in dtc\2019-06-17. A definition for each view specification in the grid is described in the following chapters.
## 7.1 Descriptions of Domains and Model Kinds

### Table 7:1 - Definitions for the Domains

<table>
<thead>
<tr>
<th>Domain</th>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metadata</td>
<td>Md</td>
<td>Identifies the metadata required to develop a suitable architecture that is fit for its purpose.</td>
</tr>
<tr>
<td>Strategic</td>
<td>St</td>
<td>Capability management process. Describes the capability taxonomy, composition, dependencies and evolution.</td>
</tr>
<tr>
<td>Operational</td>
<td>Op</td>
<td>Illustrates the Logical Architecture of the enterprise. Describes the requirements, operational behavior, structure, and exchanges required to support (exhibit) capabilities. Defines all operational elements in an implementation/solution independent manner.</td>
</tr>
<tr>
<td>Services</td>
<td>Sv</td>
<td>The Service-Orientated View (SOV) is a description of services needed to directly support the operational domain as described in the Operational View. A service within MODAF is understood in its broadest sense, as a unit of work through which a provider provides a useful result to a consumer. DoD/DoDAF: The Service Views within the Services Viewpoint describe the design for service-based solutions to support operational development processes (JCIDS) and Defense Acquisition System or capability development within the Joint Capability Areas.</td>
</tr>
<tr>
<td>Personnel</td>
<td>Pr</td>
<td>Defines and explores organizational resource types. Shows the taxonomy of types of organizational resources as well as connections, interaction and growth over time.</td>
</tr>
<tr>
<td>Resources</td>
<td>Rs</td>
<td>Captures a solution architecture consisting of resources, e.g., organizational, software, artifacts, capability configurations, and natural resources that implement the operational requirements. Further design of a resource is typically detailed in SysML or UML.</td>
</tr>
<tr>
<td>Security</td>
<td>Sc</td>
<td>Security assets and security enclaves. Defines the hierarchy of security assets and asset owners, security constraints (policy, laws, and guidance) and details where they are located (security enclaves).</td>
</tr>
<tr>
<td>Projects</td>
<td>Pj</td>
<td>Describes projects and project milestones, how those projects deliver capabilities, the organizations contributing to the projects and dependencies between projects.</td>
</tr>
<tr>
<td>Standards</td>
<td>Sd</td>
<td>MODAF: Technical Standards Views are extended from the core DoD/DoDAF views to include non-technical standards such as operational doctrine, industry process standards, etc. DoD/DoDAF: The Standards Views within the Standards Viewpoint are the set of rules governing the arrangement, interaction, and interdependence of solution parts or elements.</td>
</tr>
<tr>
<td>Actual</td>
<td>Ar</td>
<td>The analysis, e.g., evaluation of different alternatives, what-if, trade-offs, V&amp;V on the actual resource configurations. Illustrates the expected or achieved actual resource configurations.</td>
</tr>
<tr>
<td>Resources</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 7.2 - Definitions of the Model Kinds

<table>
<thead>
<tr>
<th>Model Kind</th>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Taxonomy</td>
<td>Tx</td>
<td>Presents all the elements as a standalone structure. Presents all the elements as a specialization hierarchy, provides a text definition for each one and references the source of the element.</td>
</tr>
<tr>
<td>Structure</td>
<td>Sr</td>
<td>Describes the definitions of the dependencies, connections, and relationships between the different elements.</td>
</tr>
<tr>
<td>Connectivity</td>
<td>Cn</td>
<td>Describes the connections, relationships, and interactions between the different elements.</td>
</tr>
<tr>
<td>Processes</td>
<td>Pr</td>
<td>Captures activity based behavior and flows. It describes activities, their Inputs/Outputs, activity actions and flows between them.</td>
</tr>
<tr>
<td>States</td>
<td>St</td>
<td>Captures state-based behavior of an element. It is a graphical representation of states of a structural element and how it responds to various events and actions.</td>
</tr>
<tr>
<td>Interaction</td>
<td>Is</td>
<td>Expresses a time ordered examination of the exchanges as a result of a particular scenario. Provides a time-ordered examination of the exchanges between participating elements as a result of a particular scenario.</td>
</tr>
<tr>
<td>Scenarios</td>
<td>Is</td>
<td>Expresses a time ordered examination of the exchanges as a result of a particular scenario. Provides a time-ordered examination of the exchanges between participating elements as a result of a particular scenario.</td>
</tr>
<tr>
<td>Information</td>
<td>If</td>
<td>Address the information perspective on operational, service, and resource architectures. Allows analysis of an architecture’s information and data definition aspect, without consideration of implementation specific issues.</td>
</tr>
<tr>
<td>Constraints</td>
<td>Ct</td>
<td>Details the measurements that set performance requirements constraining capabilities. Also defines the rules governing behavior and structure.</td>
</tr>
<tr>
<td>Roadmap</td>
<td>Rm</td>
<td>Addresses how elements in the architecture change over time. Also, how at different points in time or different periods of time.</td>
</tr>
<tr>
<td>Traceability</td>
<td>Tr</td>
<td>Describes the mapping between elements in the architecture. This can be between different viewpoints within domains as well as between domains. It can also be between structure and behaviors.</td>
</tr>
</tbody>
</table>
7.2 Domain Interrelationships

Although the grid is the primary means of expressing the relationship between the Domains, Model Kinds and View Specifications, because of its two-dimensional nature it is not adequate to explain the abstract interrelationships that exist between the domains. The following diagram is an indication of how the domains are interrelated.

Figure 7.2 - Domain Interrelationships

Where a Domain is shown vertically, the intent is to show that the Domain is a cross-cutting concern that goes across the levels of abstraction in the architecture.

Where a Domain is shown horizontally, the intent is to show that the Domain exists in a layer of abstraction between the Domains above and below it and there is an interrelationship with the Domains either side of it.

7.3 Domain Metamodel Diagram Legend

Note that the diagrams rely on color to aid the reader in understanding the model. Please refer to the legend below to understand the diagrams.

The following is the legend of element colors used in the DMM and what they denote.
Figure 7.3 - Legend of color codes for element types defined in UAF

The meaning of the element types in the UAF are based upon concepts put forth in the International Defence Enterprise Architecture Specification (IDEAS).

- An Individual denotes a single instance of an element
- A Type denotes a set of Individuals
- A Tuple denotes a relationship that exists between elements
- An Abstract denotes that the element has no direct use but is a means of construction
- An Enumeration is a complete, ordered listing of all the items in a collection
- An External Type is an element that exists outside of the core DMM but is referencable by elements in the DMM
8. Domain Metamodel Diagrams

Note that the diagrams rely on color to aid the reader in understanding the model. Please refer to the legend in the various diagrams to understand the specific definitions.

8.1 View Specifications

This section documents each of the view specifications of UAF.

8.1.1 View Specifications::Metadata

Stakeholders: Enterprise Architects, Technical Managers.
Concerns: architecture development process, architecture traceability, metamodel and its extensions, architecture versioning.
Definition: Identifies the metadata required to develop a suitable architecture that is fit for its purpose.

View Specifications::Metadata::Taxonomy

Stakeholders: Enterprise Architects, Technical Managers.
Concerns: metamodel and its extensions.
Definition: captures user defined metamodel extensions
Recommended Implementation: UML Profile Diagram, SysML Block Definition Diagram

View Specifications::Metadata::Structure

Stakeholders: Enterprise Architects, Technical Managers.
Concerns: domains, model kinds, and view specifications that are used to describe the architecture.
Definition: (i) lists predefined and custom domains, model kinds, and view specifications (ii) and identify the key stakeholders and their concerns.
Recommended Implementation: SysML Block Definition Diagram, SysML Package Diagram.

View Specifications::Metadata::Connectivity

Stakeholders: Enterprise Architects, people who want to understand relationships to related architectural descriptions, Technical Managers.
Concerns: high-level dependencies between architectural descriptions.
Definition: depicts and analyzes all relevant dependencies between architectural descriptions, e.g. reference architectures, as-is to to-be architectures.
Recommended Implementation: SysML Block Definition Diagram, SysML Package Diagram, matrix format.

View Specifications::Metadata::Processes

Stakeholders: Enterprise Architects, people who want to understand the architecture development process, Technical Managers.
Concerns: methodology used.
Definition: methodology used in developing the architecture.
Recommended Implementation: SysML Activity Diagram, text.

View Specifications::Metadata::States

Stakeholders: Enterprise Architects, people who want to understand the architecture governance, Technical Managers.
Concerns: architecture status.
Definition: captures version number and approval workflow of the architecture.
Recommended Implementation: SysML State Machine Diagram, state table, text.

View Specifications::Metadata::Constraints

Stakeholders: Enterprise Architects, people who want to understand constraints for the architecture, Technical Managers.
Concerns: architectural constraints.
Definition: captures assumptions and constraints on the architecture.
Recommended Implementation: tabular format, text.

**View Specifications::Metadata::Roadmap**
Stakeholders: Enterprise Architects, people who want to understand the architecture development plan, Technical Managers.
Concerns: architecture release schedule.
Definition: captures project timeline for the architecture.
Recommended Implementation: timeline, text.

**View Specifications::Metadata::Traceability**
Stakeholders: Enterprise Architects, people who want to understand impact of change across the architecture supporting assets, Technical Managers.
Concerns: reuse of architectures.
Definition: shows references to asset libraries, legacy architectures, and external sources, e.g., documents.
Recommended Implementation: SysML Block Definition Diagram, SysML Package Diagram, tabular format.

8.1.2 **View Specifications::Strategic**
Stakeholders: Capability Portfolio Managers.
Concerns: capability management process.
Definition: describe capability taxonomy, composition, dependencies and evolution.

**View Specifications::Strategic::Taxonomy**
Contains the diagrams that document the Strategic Taxonomy Viewpoint.

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

**Introduction**

1.1 **Overview**
This Appendix describes the Unified Architecture Framework, the Domain Meta-Model (DMM) that captures the concepts, relationships, and viewpoints that specify the Unified Architecture Framework Profile (UAFP). As well as providing the DMM for the UAFP, it is intended to provide a non-implementation specific metamodel for those non-UML or SysML tool vendors who may wish to implement the UAF, consequently it is not necessary to generate XMI for the UAF.

Due to the complexity of managing the multiple viewpoints with overlapping concerns and metamodels, the standard viewpoints are refactored as described in the donor frameworks into a more manageable format. This decision led to the development of the grid below.

The grid is a way of showing how the various viewpoints correspond to the generic layers of abstraction or domains (horizontal rows) and the types of model kinds or architectural representations (the columns) that describe the viewpoints. The intent of the grid is not to be complete, but to capture the information that is present in the frameworks that contribute to the UAF/P, consequently, some gaps are evident.

Notes:
- These viewpoints are not defined as part of the UAF, but are architectural artifacts that contribute to the success in defining and developing an architecture.
b. To be able to evaluate architecture behavior and constraints (i.e., non-functional requirements) it is necessary to define actual instances of the architectural elements. The expectation is that tool vendors intending to implement the UAF/P have capabilities native to their tools to enable behavioral simulation and the evaluation of measures and constraints through parametric diagrams or a proprietary equivalent.

c. The information model is a column across the abstraction layers that can be defined in any of its forms, i.e., Conceptual, Logical, or as a schema at any level of abstraction. The expectation is that most developers of the information model will use the Conceptual or Logical forms of the data model when using an abstract modeling tool.

d. The parameters column captures the measures and environments across the architecture in all the different layers of abstraction.

e. The expectation is that the physical schema model not be developed in the framework but any tool implementing the framework provides a means to import or link to representations of the physical model such as XML schemas.

f. The Metadata Taxonomy viewpoint provides a placeholder for a means to extend the profile to other domains, consequently there is not a specific diagramming type for Metadata Taxonomy.

The detailed mapping between the viewpoints shown in the grid and the viewpoints from the donor frameworks is described in the OMG document (C4i:2016-02-04). A definition for each cell follows the grid in the sections below.
### 1.2 View Type

#### Table 1.1—Definitions for the View Type

<table>
<thead>
<tr>
<th>Architectural Representation</th>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Taxonomy</td>
<td>Tx</td>
<td>Presents all the elements as a standalone structure. Presents all the elements as a specialization hierarchy, provides a text definition for each one and references the source of the element.</td>
</tr>
<tr>
<td>Structure</td>
<td>Sr</td>
<td>Describes the definitions of the dependencies, connections, and relationships between the different elements.</td>
</tr>
<tr>
<td>Connectivity</td>
<td>Cn</td>
<td>Describes the connections, relationships, and interactions between the different elements.</td>
</tr>
</tbody>
</table>
Processes

Captures activity-based behavior and flows. It describes activities, their Inputs/Outputs, activity actions and flows between them.

States

Captures state-based behavior of an element. It is a graphical representation of states of a structural element and how it responds to various events and actions.

Interaction Scenarios

Expresses a time-ordered examination of the exchanges as a result of a particular scenario. Provides a time-ordered examination of the exchanges between participating elements as a result of a particular scenario.

Information

Address the information perspective on operational, service, and resource architectures. Allows analysis of an architecture’s information and data definition aspect, without consideration of implementation-specific issues.

Constraints

Details the measurements that set performance requirements constraining capabilities. Also defines the rules governing behavior and structure.

Roadmap

Addresses how elements in the architecture change over time. Also, how at different points in time or different periods of time.

Traceability

Describes the mapping between elements in the architecture. This can be between different viewpoints within domains as well as between domains. It can also be between structure and behaviors.

<table>
<thead>
<tr>
<th>Layer of Abstraction</th>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metadata</td>
<td>Md</td>
<td>Identifies the metadata required to develop a suitable architecture that is fit for its purpose. Captures meta-data relevant to the entire architecture. Provides information pertinent to the entire.</td>
</tr>
<tr>
<td>Strategic</td>
<td>St</td>
<td>Capability management process. Describes the capability taxonomy, composition, dependencies, and evolution.</td>
</tr>
<tr>
<td>Operational</td>
<td>Op</td>
<td>Illustrates the Logical Architecture of the enterprise. Describes the requirements, operational behavior, structure, and exchanges required to support (exhibit) capabilities. Defines all operational elements in an implementation/solution independent manner.</td>
</tr>
<tr>
<td>Services</td>
<td>Sv</td>
<td>The Service-Orientated View (SOV) is a description of services needed to directly support the operational domain as described in the Operational View. A service within MODAF is understood in its broadest sense, as a unit of work through which a provider provides a useful result to a consumer. DoDAF: The Service Views within the Services Viewpoint describe the design for service-based solutions to support operational development processes (JCIDS) and Defense Acquisition System or capability development within the Joint Capability Areas.</td>
</tr>
</tbody>
</table>

Table 1.2 - Definitions for the Domains
Personnel  Pr  Defines and explores organizational resource types. Shows the taxonomy of types of organizational resources as well as connections, interaction, and growth over time.

Resources  Rs  Captures a solution architecture consisting of resources, e.g., organizational, software, artifacts, capability configurations, and natural resources that implement the operational requirements. Further design of a resource is typically detailed in SysML or UML.

Security  Se  Security assets and security enclaves. Defines the hierarchy of security assets and asset owners, security constraints (policy, laws, and guidance) and details where they are located (security enclaves).

Projects  Pj  Describes projects and project milestones, how those projects deliver capabilities, the organizations contributing to the projects and dependencies between projects.

Standards  Sd  MODAF: Technical Standards Views are extended from the core DoDAF views to include non-technical standards such as operational doctrine, industry process standards, etc. DoDAF: The Standards Views within the Standards Viewpoint are the set of rules governing the arrangement, interaction, and interdependence of solution parts or elements.

Actual Resources  Ar  The analysis, e.g., evaluation of different alternatives, what-if, trade-offs, V&V, on the actual resource configurations. Illustrates the expected or achieved actual resource configurations.

1.4 Domain Metamodel (DMM) Diagrams

This Annex comprises of various diagrams that document the Domain Metamodel (DMM) that document the MoDAF 1.5 and MoDAF 1.2 integrated model. This model was used as a basis for creating the UPDM profile.

Note that the diagrams rely on color to aid the reader in understanding the model. Please refer to the legend below to understand the diagrams.

The following is the legend of element colors used in the DMM and what they denote.
The meaning of the element types in the UAF are based upon concepts put forth in the International Defence Enterprise Architecture Specification (IDEAS).

- An Individual denotes a single instance of an element
- A Type denotes a set of Individuals
- A Tuple denotes a relationship that exists between elements
- An Abstract denotes that the element has no direct use but is a means of construction
- An Enumeration is a complete, ordered listing of all the items in a collection
- An External Type is an element that exists outside of the core DMM but is referencable by elements in the DMM
2 View Specifications

This section documents each of the view specifications of UAF.

2.1 View Specifications::Strategic

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

2.1.1 View Specifications::Strategic::Taxonomy

Contains the diagrams that document the Strategic Taxonomy Viewpoint.

2.1.1.1 View Specifications::Strategic::Taxonomy::Strategic Taxonomy

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

Figure 2.1 - Strategic Taxonomy

Elements

- Capability
- CapabilityGeneralization
- CapabilityRole
8.1.22.1.2 View Specifications::Strategic::Structure

Contains the diagrams that document the Strategic Structure Viewpoint.
2.1.2.1 View Specifications::Strategic::Structure::Strategic Structure

Stakeholders: PMs, Enterprise Architects, Executives
Concerns: capability needs
Definition: shows the relationship between EnterprisePhases and the Capabilities that are intended to be developed during the enterprise phases, and the organizations involved in the enterprise.
Recommended Implementation: SysML Block Definition Diagram
Figure 2.2 - Strategic Structure

Elements

- ActualEnduringTask
- ActualEnterprisePhase
- ActualOrganization
- ActualResponsibleResource
- Capability
- CapableElement
- EnduringTask
- EnterpriseGoal
- EnterprisePhase
- EnterpriseVision
- Exhibits
- OperationalArchitecture
- OrganizationInEnterprise
- ResourceArchitecture
- StructuralPart
- TemporalPart
- WholeLifeEnterprise
- ActualEnduringTask
- ActualEnterprisePhase
- ActualOrganization
- ActualResponsibleResource
- Capability
UAF-19 (10) replace definition of Strategic Structure

Unified Architecture Framework (UAF), v1.0
2.1.3 View Specifications::Strategic::Connectivity
Contains the diagrams that document the Strategic Connectivity Viewpoint.

2.1.3.1 View Specifications::Strategic::Connectivity::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 2.3 - Strategic Connectivity

Elements
- Capability
- CapabilityDependency
- CapabilityRole
- CapabilityRoleDependency
2.1.4 View Specifications::Strategic::States
Contains the diagrams that document the Strategic States Viewpoint.

2.1.4.1 View Specifications::Strategic::States::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 2.4 - Strategic States

Elements

- AchievedEffect
- Achiever
- ActualOrganization
- ActualOrganizationalResource
- ActualPerson
- ActualPost
- ActualResource
- ActualResponsibleResource
- ActualState
- Capability
- CapabilityConfiguration
- DesiredEffect
- Desirer
2.1.5 View Specifications::Strategic::Constraints
Contains the diagrams that document the Strategic Constraints Viewpoint.

2.1.5.1 View Specifications::Strategic::Constraints::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

![Diagram of Capability, PropertySet, and Measurement]

Figure 2.5 - Strategic Constraints
• Capability
• Measurement
• PropertySet
2.1.6 View Specifications::Strategic::Roadmap
Contains the diagrams that document the Strategic Roadmap Viewpoint.

2.1.6.1 View Specifications::Strategic::Roadmap::Deployment 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 2.6 - Strategic Roadmap: Deployment
• \texttt{ActualEnterprisePhase}
• \texttt{ActualOrganization}
• \texttt{ActualPerson}
• \texttt{ActualPost}
• \texttt{ActualProject}
• \texttt{ActualProjectMilestone}
• \texttt{ActualResource}
• \texttt{ActualResponsibleResource}
• \texttt{Capability}
• \texttt{CapabilityConfiguration}
2.1.6.2 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 2.7 - Strategic Roadmap: Phasing

Elements

- ActualProject
- ActualProjectMilestone
- ActualResource
2.1.7 View Specifications::Strategic::Traceability

Contains the diagrams that document the Strategic Traceability Viewpoint.

2.1.7.1 View Specifications::Strategic::Traceability::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 2.8 - Strategic Traceability

Elements

- Activity
- ActualEnduringTask
- Capability
- CapabilityForTask
- Function
- Implements
- MapsToCapability
- OperationalActivity
- Process

- StandardOperationalActivity
2.2 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.

2.2.1 View Specifications::Operational::Taxonomy

Contains the diagrams that document the Operational Taxonomy Viewpoint.

2.2.1.1 View Specifications::Operational::Taxonomy::Operational Taxonomy

Stakeholders: Business Architects, Systems Engineers, Enterprise Architects, Owners responsible for Operational Agents.

Concerns: OperationalAgent types
Definition: shows the taxonomy of types of OperationalAgents
Recommended Implementation: SysML Block Definition Diagram, SysML Internal Block Diagram.
Figure 2.9 - Operational Taxonomy

Elements

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

**Asset**
- CapabilityConfiguration
- ConceptItem

---

2 UAF-19 (2) modify OperationalPerformer owners to Owners responsible for OperationalPerformers

14 Unified Architecture Framework (UAF). v1.0
2.2.2 View Specifications::Operational::Structure

Contains the diagrams that document the Operational Structure Viewpoint.

2.2.2.1 View Specifications::Operational::Structure::Operational Structure

Stakeholders: Business Architects, Systems Engineers, Enterprise Architects, Owners responsible for Operational Agents.
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.

Stakeholders: Business Architects, Systems Engineers, Enterprise Architects, Owners responsible for Operational Performers.
Concerns: identifies the operational exchange requirements between Operational Performers.
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.

---

3 Modify text from OperationalPerformer Owners to Owners responsible for OperationalPerformers.
4 UAF-19 (4) delete nodes and change to OperationalPerformers.
Figure 2.10 - Operational Structure

Elements
- ActualEnvironment
- ActualLocation
- Asset
- Capability
• CapableElement
• Exhibits
• IsCapableToPerform
• KnownResource
• LocationHolder
• OperationalActivity
• OperationalAgent
• OperationalArchitecture
• OperationalAsset
• OperationalPerformer
• OperationalRole
• ProblemDomain
• ActualLocation
• Asset
• Capability
• CapableElement
• Environment
• Exhibits
• IsCapableToPerform
• KnownResource
• LocationHolder
• OperationalActivity
• OperationalAgent
• OperationalArchitecture
• OperationalConnector
• OperationalExchange
• OperationalExchangeItem
• OperationalInterface
• OperationalMethod
• OperationalParameter
• OperationalPerformer
• OperationalPort
• OperationalRole
• ProblemDomain
2.2.3 View Specifications::Operational::Connectivity

Contains the diagrams that document the Operational Connectivity Viewpoint.

2.2.3.1 View Specifications::Operational::Connectivity::Operational Connectivity

Stakeholders: Systems Engineers, Architects, Solution Providers

Concerns: captures the interfaces between Operational Performers

Definition: summarizes the logical exchanges between Operational Performers of information, systems, personnel, energy, etc. and the logical activities that produce and consume them. Measurements can optionally be included.

Recommended Implementation: SysML Internal Block Diagram, tabular format

Figure 2.11 - Operational Connectivity

---

3 UAF-19 items 5 and 6 replace terms logical nodes and node with Operational Performers
6 UAF-2, UAF-21
Elements

- CapabilityConfiguration
- Exchange
- GeoPoliticalExtentType
- InformationElement
- IsCapableToPerform
- MeasurableElement
- MeasurementSet
- NaturalResource
- OperationalActivity
- OperationalActivityAction
- OperationalActivityEdge
- OperationalAgent
- OperationalConnector
- OperationalExchange
- OperationalExchangeItem
- OperationalInterface
- OperationalPerformer
- OperationalPort
- OperationalRole
  - OperationalSignal
- OrganizationalResource
- PhysicalResource
- ResourceArchitecture
- ResourceArtifact
- ResourcePerformer

2.2.4 View Specifications::Operational::Processes

Contains the diagrams that document the Operational Processes Viewpoint.

2.2.4.1 View Specifications::Operational::Processes::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 2.12 - Operational Processes

Elements

- Activity
- ActivityPerformableUnderCondition
- ActualCondition
- ActualMeasurementSet
- ActualService
- IsCapableToPerform
- MeasurableElement
- OperationalActivity
- OperationalActivityAction
- OperationalActivityEdge
- OperationalAgent
- OperationalExchange
- OperationalExchangeItem
- OperationalMethod
- OperationalParameter
- OperationalPerformer
- OperationalRole
- PerformsInContext
- Process
- ProcessEdge
- ProcessOperation
- ProcessParameter
- ProcessUsage
- RequiredServiceLevel
- ServiceSpecification
- StandardOperationalActivity
- UML2.5Metamodel::Activity
- UML2.5Metamodel::ActivityEdge
- UML2.5Metamodel::CallBehaviorAction
- UML2.5Metamodel::Operation
- UML2.5Metamodel::Parameter
- ActivityPerformableUnderCondition
- ActualEnduringTask
- ActualMeasurementSet
- ActualService
- Condition
- EnduringTask
- Implements
- IsCapableToPerform
- MeasurableElement
- OperationalActivity
- OperationalActivityAction
- OperationalActivityEdge
- OperationalAgent
- OperationalExchange
- OperationalExchangeItem
- OperationalMethod
- OperationalParameter
**View Specifications::Operational::Processes::Operational Processes BPMN Semantics**

**Stakeholders:** Business Architect, Enterprise Architects.

**Concerns:** captures activity based behavior and flows using BPMN notation.

**Definition:** describes the BPMN processes 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 using BPMN notation.

**Recommended Implementation:** BPMN Process Diagram.

---

**Elements**
- ActivityPerformableUnderCondition
- ActualCondition
- ActualEnduringTask
- ActualMeasurementSet
- ActualService
- AssetRole
- BPMN2Metamodel::BPMNMessage
- BPMN2Metamodel::CallActivity
- BPMN2Metamodel::MessageFlow
- BPMN2Metamodel::ResourceRole
- BPMN2Metamodel::Process
- BPMN2Metamodel::EnduringTask
- BPMN2Metamodel::SequenceFlow
- OperationalAgent
- OperationalPerformer
- OperationalActivity
- OperationalActivityAction
- OperationalActivityEdge
- OperationalExchange
- OperationalExchangeItem
- OperationalMethod
- OperationalParameter
- OperationalPerformer
- OperationalRole
- OperationalRole
- Process
- ProcessEdge
- ServiceSpecification

---

**Figure 8:13 - Operational Processes BPMN Semantics**
2.2.5 View Specifications::Operational::States

Contains the diagrams that document the Operational States Viewpoint.

2.2.5.1 View Specifications::Operational::States::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 Machine Diagram, SysML State Diagram
2.2.6 View Specifications::Operational::Interaction Scenarios

Contains the diagrams that document the Operational Interaction Scenarios Viewpoint.

2.2.6.1 View Specifications::Operational::Interaction Scenarios::Operational Interaction Scenario

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.

Recommended Implementation: SysML Sequence Diagram, BPMN Collaboration Diagram
Elements

- **Activity**
- **InteractionMessage**
- **InteractionRole**
- **InteractionScenario**
- **OperationalActivity**

Unified Architecture Framework (UAF), v1.0
2.2.7 View Specifications::Operational::Constraints

Contains the diagrams that document the Operational Constraints Viewpoint.

2.2.7.1 View Specifications::Operational::Constraints::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 2.15 - Operational Constraints

Elements
- ConstraintBlock
- DataModel
- InformationElement
- OperationalActivity
- OperationalAgent
- OperationalConstraint
- OperationalExchange
2.2.8 View Specifications::Operational::Traceability

Contains the diagrams that document the Operational Traceability Viewpoint.

2.2.8.1 View Specifications::Operational::Traceability::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
2.3 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

Unified Architecture Framework (UAF), v1.0
Capability or to support an Operational Activity.

2.3.1 View Specifications::Services::Taxonomy

Contains the diagrams that document the Services Taxonomy Viewpoint.

2.3.1.1 View Specifications::Services::Taxonomy::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 2.17 - Services Taxonomy

Elements

- ActualMeasurement
- ActualMeasurementSet
- ActualService
- Measurement
- PropertySet
- ProvidedServiceLevel
- RequiredServiceLevel
- ServicePolicy
- ServiceSpecification
- ServiceSpecificationGeneralization
2.3.2 View Specifications::Services::Structure
Contains the diagrams that document the Services Structure Viewpoint.

2.3.2.1 View Specifications::Services::Structure::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 2.18 - Services Structure

Elements

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

2.3.3 View Specifications::Services::Connectivity
Contains the diagrams that document the Services Connectivity Viewpoint.

2.3.3.1 View Specifications::Services::Connectivity::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 2.19 - Services Connectivity

Elements

- ActualMeasurementSet
- ActualService
- MeasurableElement
- ProvidedServiceLevel
- RequiredServiceLevelServiceConnector
- ServiceInterface
- ServiceMethod
- ServiceParameter
- ServicePort
- ServiceSpecification
2.3.4 View Specifications::Services::Processes
Contains the diagrams that document the Services Processes Viewpoint.

2.3.4.1 View Specifications::Services::Processes::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 Activity Diagram, SysML Block Definition Diagram, SysML Block Definition Diagram, SysML Internal Block Diagram, tabular format
Figure 2.20 - Services Processes

Elements

- IsCapableToPerform
- Process
- ProcessEdge
- ProcessOperation
- ProcessParameter
- ProcessUsage
- ServiceFunction
- ServiceFunctionAction
- ServiceFunctionEdge
- ServiceMethod
**View Specifications::Services::Processes::Services Processes BPMN Semantics**

*Stakeholders: Solution Providers, 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 using BPMN.*

*Recommended Implementation: BPMN Process Diagram, SysML Block Definition Diagram.*

---

**Figure 8:22 - Services Processes BPMN Semantics**

**Elements**

- BPMN2Metamodel::CallActivity
- BPMN2Metamodel::Process
- BPMN2Metamodel::ResourceRole
- BPMN2Metamodel::SequenceFlow
- InteractionRole
- IsCapableToPerform

---

Unified Architecture Framework (UAF), v1.0
2.3.5 View Specifications::Services::States

Contains the diagrams that document the Services States Viewpoint.

2.3.5.1 View Specifications::Services::States::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.

![Services States Diagram](image)

Figure 2.21 - Services States
Elements

- **ServiceSpecification**
- **ServiceStateDescription**
- **StateDescription**
- **UML2.5Metamodel::StateMachine**

### 2.3.6 View Specifications::Services::Interaction Scenarios

Contains the diagrams that document the Services Interaction Scenarios Viewpoint.

#### 2.3.6.1 View Specifications::Services::Interaction Scenarios::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 2.22 - Services Interaction Scenarios

Elements

- InteractionMessage
- InteractionRole
- InteractionScenario
- ServiceFunction
- ServiceInteractionScenario
- ServiceMessage
- ServiceMethod
- ServiceSpecification
2.3.7 View Specifications::Services::Constraints

Contains the diagrams that document the Services Constraints Viewpoint.
2.3.7.1 View Specifications::Services::Constraints::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 2.23 - Services Constraints

Elements
- ConstraintBlock
- Rule
- ServicePolicy
- ServiceSpecification

2.3.8 View Specifications::Services::Roadmap

Contains the diagrams that document the Services Roadmap Viewpoint.

2.3.8.1 View Specifications::Services::Roadmap::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
Elements

- ActualProject
- ActualProjectMilestone
- MilestoneDependency
- ServiceSpecification
- VersionedElement
- VersionOfConfiguration
2.3.9 View Specifications::Services::Traceability

Contains the diagrams that document the Services Traceability Viewpoint.

2.3.9.1 View Specifications::Services::Traceability::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: tabular or matrix format, timeline, SysML Block Definition Diagram, SysML Internal Block Diagram
2.4 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).

2.4.1 View Specifications::Personnel::Taxonomy
Contains the diagrams that document the Personnel Taxonomy Viewpoint.

2.4.1.1 View Specifications::Personnel::Taxonomy::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
Elements

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

2.4.2 View Specifications::Personnel::Structure

Contains the diagrams that document the Personnel Structure Viewpoint.

2.4.2.1 View Specifications::Personnel::Structure::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 2.27 - Personnel Structure

Elements

Unified Architecture Framework (UAF), v1.0
2.4.3 View Specifications::Personnel::Connectivity

Contains the diagrams that document the Personnel Connectivity Viewpoint.
2.4.3.1 View Specifications::Personnel::Connectivity::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: SysML Internal Block Diagram, table tabular format
Figure 2.28 - Personnel Connectivity

Elements

- Command
- Control
- DataElement
- Environment
- Exchange
- Function
- IsCapableToPerform
- MeasurableElement
- Measurement
- MeasurementSet
- Organization
- OrganizationalResource
- Person
- PhysicalResource
- Post
- PropertySet
- Resource
- ResourceConnector
- ResourceExchange
- ResourceExchangeItem
- ResourceInterface
- ResourcePerformer
- ResourcePort
- ResourceRole
- Responsibility
- Activity
- Command
- Control
- DataElement
- Environment
- Exchange
- Function
- FunctionAction
2.4.4 View Specifications::Personnel::Processes

Contains the diagrams that document the Personnel Processes Viewpoint.

2.4.4.1 View Specifications::Personnel::Processes::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, as described in the Resources Processes section.
Figure 2.29 - Personnel Processes

Elements

- ActivityPerformableUnderCondition
- ActualCondition
- DataElement
- Function
- FunctionAction
- FunctionEdge
- IsCapableToPerform
- Organization
- OrganizationalResource
- PerformsInContext
- PhysicalResource
- Post
- Process
- ProcessEdge
- ProcessOperation
- ProcessParameter
- ProcessUsage
• ResourceExchange
• ResourceExchangeItem
• ResourceMethod
• ResourceParameter
• ResourcePerformer
• ResourceRole
• Responsibility
• UML2.5Metamodel::Activity
• UML2.5Metamodel::ActivityEdge
• UML2.5Metamodel::CallBehaviorAction
• UML2.5Metamodel::Operation
• UML2.5Metamodel::Parameter
• Activity
• ActivityPerformableUnderCondition
• Condition
• DataElement
• Function
• FunctionAction
• FunctionEdge
• Implements
• IsCapableToPerform
• OperationalActivity
• Organization
• OrganizationalResource
• PerformsInContext
2.4.5 View Specifications::Personnel::States

Contains the diagrams that document the Personnel States Viewpoint.

2.4.5.1 View Specifications::Personnel::States::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 Machine Diagram, SysML State Diagram
Figure 2.30 - Personnel States

Elements

- Organization
- OrganizationalResource
- Person
- PhysicalResource
- Post
- ResourcePerformer
- ResourceStateDescription
• Responsibility
  • StateDescription
  • UML2.5Metamodel::StateMachine

2.4.6 View Specifications::Personnel::Interaction Scenarios
Contains the diagrams that document the Personnel Interaction Scenarios Viewpoint.

2.4.6.1 View Specifications::Personnel::Interaction Scenarios::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 2.31 - Personnel Interaction Scenarios
2.4.7 View Specifications::Personnel::Constraints
Contains the diagrams that document the Personnel Constraints Viewpoint.

2.4.7.1 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 2.32 - Personnel Constraints: Competence

Elements

Unified Architecture Framework (UAF), v1.0
• ActualOrganization
• ActualOrganizationalResource
• ActualPerson
• ActualPost
2.4.7.2 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
2.4.7.3 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 2.34 - Personnel Constraints: Performance

Elements

- ActivityPerformableUnderCondition
- ActualCondition
- ActualMeasurement
- ActualMeasurementSet
- ActualOrganizationalResource
- ActualPerson
- ActualPost
- ActualPropertySet
- ActualResource
- ActualResponsibleResource
- ActualState
- DesiredEffect
- Desirer
- Function
- IsCapableToPerform
- MeasurableElement
- Measurement
- MeasurementSet
• Organization
• OrganizationalResource
• Person
• PhysicalResource
• Post
• Process
• ResourcePerformer
• Responsibility
• Activity
• ActivityPerformableUnderCondition
• ActualMeasurement
• ActualMeasurementSet
• ActualOrganizationalResource
• ActualPerson
• ActualPost
• ActualPropertySet
• ActualResource
• ActualResponsibleResource
• ActualState
• Condition
• DesiredEffect
• Desirer
2.4.8 View Specifications::Personnel::Roadmap

Contains the diagrams that document the Personnel Roadmap Viewpoint.

2.4.8.1 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 2.35 - Personnel Roadmap: Availability

Elements

- ActualMeasurement
- ActualOrganizationalResource
- ActualPerson
- ActualPost
- ActualProject
- ActualProjectMilestone
- ActualPropertySet
- ActualResource
- ActualResponsibleResource
- ActualState
- Asset
- FillsPost
- Measurement
- OrganizationalResource
- Person
- PhysicalResource
- Post
- Project
- ProjectMilestone
- PropertySet
- ResourceAsset
- ResourcePerformer
- ActualMeasurement
- ActualOrganizationalResource
- ActualPerson
- ActualPost
- ActualProject
- ActualProjectMilestone
- ActualPropertySet
- ActualResource
- ActualResponsibleResource
- ActualState
- Asset
2.4.8.2 View Specifications::Personnel::Roadmap::Personnel Roadmap: Evolution

Stakeholders: Human resources, Solution Providers
Concerns: organizational structure changes over time
Definition: provides an overview of how an 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 2.36 - Personnel Roadmap: Evolution

Elements

- **ActualProject**
- **ActualProjectMilestone**
- **MilestoneDependency**
- **Organization**
- **OrganizationalResource**
- **Person**
2.4.8.3 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 2.37 - Personnel Roadmap: Forecast

Elements

- ActualEnterprisePhase
- ActualPropertySet
- ActualState
- Asset
- Competence
- Forecast
- Organization
- OrganizationalResource
- Person

Unified Architecture Framework (UAF), v1.0
2.4.9 View Specifications::Personnel::Traceability

Contains the diagrams that document the Personnel Traceability Viewpoint.

2.4.9.1 View Specifications::Personnel::Traceability::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

![Personnel Traceability Diagram](image)

Figure 2.38 - Personnel Traceability

Elements

- Function
- Implements
- OperationalActivity
- ServiceFunction

2.5 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.
2.5.1 View Specifications::Resources::Taxonomy
Contains the diagrams that document the Resources Taxonomy Viewpoint.

2.5.1.1 View Specifications::Resources::Taxonomy::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 2.39 - Resources Taxonomy

Elements

- Asset
- CapabilityConfiguration
- Implements
- Measurement
- NaturalResource
- OperationalAgent
- OperationalPerformer
- Organization
- OrganizationalResource
- Person
- PhysicalResource
2.5.2 View Specifications::Resources::Structure

Contains the diagrams that document the Resources Structure Viewpoint.

2.5.2.1 View Specifications::Resources::Structure::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 Operational Performer(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, SysML Block Definition Diagram, SysML Internal Block Diagram
Figure 2.40 - Resources Structure

Elements

- Asset

Unified Architecture Framework (UAF), v1.0
2.5.3 View Specifications::Resources::Connectivity

Contains the diagrams that document the Resources Connectivity Viewpoint.

2.5.3.1 View Specifications::Resources::Connectivity::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: SysML Internal Block Diagram, tabular format, tabular format²

² UAF-2, UAF-21
Elements

- **CapabilityConfiguration**
- **DataElement**
- **Exchange**
- **Function**
- **FunctionAction**
- **FunctionEdge**
- **GeoPoliticalExtentType**
- **IsCapableToPerform**
- **MeasurableElement**
- **Measurement**
- **MeasurementSet**
- **NaturalResource**
- **Organization**
- **OrganizationalResource**
- **Person**
- **PhysicalResource**
- **Process**
- **Post**
- **PropertySet**
- **Resource**
- **ResourceArchitecture**
• ResourceArtifact
• ResourceConnector
• ResourceExchange
• ResourceExchangeItem
• ResourceInterface
• ResourceMitigation
• ResourcePerformer
• ResourcePort
• ResourceRole
• ResourceSignal
• Software
• Technology
• Activity
• CapabilityConfiguration
• Exchange
• Function
• FunctionAction
• FunctionEdge
• IsCapableToPerform
• MeasurableElement
• Measurement
• MeasurementSet
• NaturalResource
• Organization
• OrganizationalResource
2.5.4 View Specifications::Resources::Processes

Contains the diagrams that document the Resources Processes Viewpoint.

2.5.4.1 View Specifications::Resources::Processes::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 2.42 - Resources Processes

Elements

- ActivityPerformableUnderCondition
- ActualCondition
- DataElement
- Function
- FunctionAction
- FunctionEdge
- Implements
- OperationalActivity
- PerformsInContext
- PhysicalResource
- Process
- ProcessEdge
- ProcessParameter
- ProcessUsage
- ResourceArchitecture
- ResourceExchange
- ResourceExchangeItem
- ResourceParameter
- ResourcePerformer
- ResourceRole
- UML2.5Metamodel::Activity
- UML2.5Metamodel::ActivityEdge
- UML2.5Metamodel::CallBehaviorAction
- UML2.5Metamodel::Parameter

**Activity**
- ActivityPerformableUnderCondition

**Condition**
- DataElement
- Function
- FunctionAction
- FunctionEdge
- Implements
- OperationalActivity
- PerformsInContext
- PhysicalResource
- ResourceArchitecture
- ResourceExchange
- ResourceExchangeItem
- ResourceParameter
- ResourcePerformer
- ResourceRole

---

**View Specifications::Resources::Processes::Resources Processes BPMN Semantics**

Stakeholders: Solution Providers, IT Architects.

Concerns: captures activity based behavior and flows using BPMN.

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 using BPMN.

Recommended Implementation: BPMN Process Diagram.
Figure 8:45 - Resources Processes BPMN Semantics

Elements
- ActivityPerformableUnderCondition
- AssetRole
- BPMN2Metamodel::BPMNMessage
- BPMN2Metamodel::CallActivity
- BPMN2Metamodel::MessageFlow
- BPMN2Metamodel::Process
- BPMN2Metamodel::ResourceRole
- BPMN2Metamodel::SequenceFlow
- DataElement
- Exchange
- ExchangeItem
- Function
- FunctionAction
- FunctionEdge
- Implements
- OperationalActivity
- PerformsInContext
- PhysicalResource
- Process
2.5.5 View Specifications::Resources::States

Contains the diagrams that document the Resources States Viewpoint.
2.5.5.1 View Specifications::Resources::States::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 Machine Diagram

Figure 2.43 - Resources States

Elements
- ResourcePerformer
- ResourceStateDescription
- StateDescription
- UML2.5Metamodel::StateMachine

2.5.6 View Specifications::Resources::Interaction Scenarios

Contains the diagrams that document the Resources Interaction Scenarios Viewpoint.

2.5.6.1 View Specifications::Resources::Interaction Scenarios::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 2.44 - Resources Interaction Scenarios
Elements

- Function
- InteractionMessage
- InteractionRole
- InteractionScenario
- ResourceExchange
- ResourceInteractionScenario
- ResourceMessage
- ResourceMethod
- ResourcePerformer
- ResourceRole
- UML2.5Metamodel::Interaction
- UML2.5Metamodel::Lifeline
- UML2.5Metamodel::Message

2.5.7 View Specifications::Resources::Constraints

Contains the diagrams that document the Resources Constraints Viewpoint.

2.5.7.1 View Specifications::Resources::Constraints::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 2.45 - Resources Constraints
Elements

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

2.5.8 View Specifications::Resources::Roadmap

Contains the diagrams that document the Resources Roadmap Viewpoint.

2.5.8.1 View Specifications::Resources::Roadmap::Resources 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 2.46 - Resources Roadmap: Evolution
Elements

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

2.5.8.2 View Specifications::Resources::Roadmap::Resources 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 2.47 - Resources Roadmap: Forecast

Elements

- ActualEnterprisePhase
• ActualPropertySet
• ActualState
• CapabilityConfiguration
• Forecast
• NaturalResource
• PhysicalResource
• ResourceArchitecture
• ResourceArtifact
• ResourceMitigation
• ResourcePerformer
• Software
• SubjectOfForecast
• Technology

2.5.9 View Specifications::Resources::Traceability

Contains the diagrams that document the Resources Traceability Viewpoint.
2.5.9.1 View Specifications::Resources::Traceability::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 2.48 - Resources Traceability

Elements
- Capability
2.6 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.
2.6.1 View Specifications::Security::Taxonomy
Contains the diagrams that document the Security Taxonomy Viewpoint.

2.6.1.1 View Specifications::Security::Taxonomy::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 2.49 - Security Taxonomy

Elements

- **ActualLocation**
- **Asset**
• DataElement
• InformationElement
• LocationHolder
• Measurement
• MeasurementSet
• OperationalAgent
• OperationalArchitecture
• OperationalAsset
• OperationalMitigation
• OperationalPerformer
• PropertySet
• ResourceArchitecture
• ResourceAsset
• ResourceMitigation
• ResourcePerformer
• Risk
• SecurityAvailability
• SecurityCategory
• SecurityClassification
• SecurityClassificationKind
• SecurityEnclave
• SecurityIntegrity
• SecurityMeasurement
• ActualLocation
• Asset
• DataElement
• InformationElement
• LocationHolder
• Measurement
• MeasurementSet
• OperationalAgent
• OperationalPerformer
• PropertySet
• ResourceArchitecture
2.6.2 View Specifications::Security::Structure

Contains the diagrams that document the Security Structure Viewpoint.

2.6.2.1 View Specifications::Security::Structure::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 2.50 - Security Structure

Elements

- Asset
- DataElement
- DataRole
- InformationElement
- InformationRole
- OperationalAgent
- OperationalAsset
- OperationalConnector
- OperationalExchange
• OperationalPerformer
• OperationalPort
• OperationalRole
• ResourceAsset
• ResourceConnector
• ResourceExchange
• ResourcePerformer
• ResourcePort
• ResourceRole
• SecurityCategory

Asset
• DataElement
• InformationElement
• OperationalAgent
• OperationalConnector
2.6.3 View Specifications::Security::Connectivity

Contains the diagrams that document the Security Connectivity Viewpoint.

2.6.3.1 View Specifications::Security::Connectivity::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: SysML Internal Block Diagram, tabular format Recommended Implementation: tabular format
Figure 2.51 - Security Connectivity

Elements

- Caveat
- MeasurableElement
- MeasurementSet
- OperationalAgent
- OperationalConnector
- OperationalExchange
- OperationalExchangeItem
- OperationalInterface
- OperationalPerformer
- OperationalPort
- OperationalRole
- ResourceConnector
- ResourceExchange
- ResourceExchangeItem
- ResourceInterface
- ResourcePerformer
- ResourcePort
- ResourceRole
- SecurityConstraint
- SubjectOfSecurityConstraint
- Asset
- Caveat
- MeasurableElement
- MeasurementSet
- OperationalAgent
- OperationalConnector
- OperationalExchange
- OperationalExchangeItem
- OperationalInterface
- OperationalPerformer
- OperationalPort
- OperationalRole
- ResourceConnector
- ResourceExchange
- ResourceExchangeItem
- ResourceInterface
- ResourcePerformer
2.6.4 View Specifications::Security::Processes

Contains the diagrams that document the Security Processes Viewpoint.

2.6.4.1 View Specifications::Security::Processes::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, BPMN Process Diagram as described in Operational Processes and Resources Processes sections.
2.52 - Security Processes

Elements

- Function
- FunctionAction
- IsCapableToPerform
- MeasurableElement
- MeasurementSet
- OperationalActivity
- OperationalActivityAction
- OperationalAgent
- OperationalRole
- PerformsInContext
- Process
- ResourcePerformer
- ResourceRole
- SecurityProcess
- SecurityProcessAction
- Activity
- Asset
- EnhancedSecurityControl
- Enhances
- Function
- FunctionAction
• IsCapableToPerform
• MeasurableElement
• MeasurementSet
• OperationalActivity
• OperationalActivityAction
• OperationalAgent
2.6.5 View Specifications::Security::Constraints

Contains the diagrams that document the Security Constraints Viewpoint.

2.6.5.1 View Specifications::Security::Constraints::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 2.53 - Security Constraints

Elements

- ActualMeasurement
- ActualPropertySet
- ActualResource
- ActualResponsibleResource
- ActualRisk
- Affects
- AffectsInContext
- Asset
- AssetRole
- EnhancedSecurityControl
- Enhances
- Measurement
- Mitigates
- OperationalAgent
- OperationalRole
- OrganizationalResource
- OwnsRisk
- OwnsRiskInContext
- PropertySet
- Protects
- ProtectsInContext
- ResourcePerformer
- ResourceRole
- Risk
- Rule
- Satisfy
- SecurityConstraint
- SecurityControl
- SecurityControlFamily
- SecurityProcess
- SubjectOfSecurityConstraint

  ActualMeasurement
  - ActualPropertySet
  - ActualResource
  - ActualResponsibleResource
  - ActualRisk
  - Affects
  - Asset
  - AssetRole
  - Measurement
  - OperationalAgent
  - OperationalRole
  - PropertySet
  - ResourcePerformer
2.6.6 View Specifications::Security::Traceability

Contains the diagrams that document the Security Traceability Viewpoint.

2.6.6.1 View Specifications::Security::Traceability::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 2.54 - Security Traceability

Elements

- Affects
- AffectsInContext
- Asset
- AssetRole
- DataRole
- InformationRole
- Mitigates

Unified Architecture Framework (UAF), v1.0
- OperationalRole
- OwnsRiskInContext
- Protects
- ProtectsInContext
- ResourceRole
- Risk
- Satisfy
- SecurityControl
- Affects
- AssetRole
- Mitigates
- OperationalRole
- OwnsRisk
- ResourceRole
- Risk
- SecurityProperty
2.7 View Specifications::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.

2.7.1 View Specifications::Project::Taxonomy

Contains the diagrams that document the Project Taxonomy Viewpoint.

2.7.1.1 View Specifications::Project::Taxonomy::Project 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
Elements

- `ActualProject`
- `ActualProjectMilestone`
- `MilestoneDependency`
- `Project`
- `ProjectMilestone`
- `ProjectMilestoneRole`
- `ProjectSequence`
2.7.2 View Specifications::Project::Structure
Contains the diagrams that document the Project Structure Viewpoint.

2.7.2.1 View Specifications::Project::Structure::Project 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 2.56 - Project Structure

Elements

- ActualOrganization
- ActualPost
- ActualProject
- ActualPropertySet
- ActualResponsibleResource
- ActualState
- Project
- ProjectMilestone
- ProjectMilestoneRole
- ProjectTheme
2.7.3 View Specifications::Project::Connectivity

Contains the diagrams that document the Project Connectivity Viewpoint.

2.7.3.1 View Specifications::Project::Connectivity::Project 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 2.57 - Project Connectivity

Elements

- Project
- ProjectMilestone
- ProjectMilestoneRole
- ResourcePerformer
- ResponsibleFor
- StatusIndicators
• ResourcePerformer

---

8 UAF-43 add hyperlinked ResourceRole to elements list
2.7.4 View Specifications::Project::Processes
Contains the diagrams that document the Project Processes Viewpoint.

2.7.4.1 View Specifications::Project::Processes::Project Processes
Stakeholders: PMs
Concerns: captures project tasks (ProjectActivities) and flows between them
Definition: describes the ProjectActivities that are normally conducted in the course of projects to support capability(ies) and implement resources. It describes the ProjectActivities, their Inputs/Outputs, ProjectActivityActions and flows between them. Recommended Implementation: SysML Activity Diagram, SysML Block Definition Diagram, BPMN Process Diagram as described in Resources Processes section.
Figure 2.58 - Project Processes
Elements

- ActualProject
- DataElement
- Function
- FunctionAction
- FunctionEdge
- GeoPoliticalExtentType
- IsCapableToPerform
- Organization
2.7.5 View Specifications::Project::Roadmap

Contains the diagrams that document the Project Roadmap Viewpoint.

2.7.5.1 View Specifications::Project::Roadmap::Project 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 2.59 - Project Roadmap

Elements

- ActualProject
- ActualProjectMilestone
- ActualProjectMilestoneRole
- ActualPropertySet
- ActualResource
- ActualState
- CapabilityConfiguration
- FieldedCapability
- MilestoneDependency
- Project
- ProjectMilestone
- ProjectMilestoneRole
- ProjectSequence
- ProjectStatus
- ProjectTheme
- ResourceArchitecture
- ResourcePerformer
- StatusIndicators
2.7.6 View Specifications::Project::Traceability
Contains the diagrams that document the Project Traceability Viewpoint.

2.7.6.1 View Specifications::Project::Traceability::Project 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 2.60 - Project Traceability

Elements
- ActualProject
- ActualProjectMilestone
- ActualResource
- ResourcePerformer
- CapableElement
- Capability
- Exhibits

Unified Architecture Framework (UAF), v1.0
2.8 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.
2.8.1 View Specifications::Standards::Taxonomy
Contains the diagrams that document the Standards Taxonomy Viewpoint.

2.8.1.1 View Specifications::Standards::Taxonomy::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 2.61 - Standards Taxonomy

Elements

- ActualOrganization
- CapabilityConfiguration
- Protocol
- ProtocolStack
- ResourceArchitecture
- ResourcePerformer
- Standard
- StandardOperationalActivity
- UAFAElement
2.8.2 View Specifications::Standards::Structure
Contains the diagrams that document the Standards Structure Viewpoint.

2.8.2.1 View Specifications::Standards::Structure::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 2.62 - Standards Structure

Elements
- Protocol
- ProtocolLayer
- ProtocolStack
- Standard

2.8.3 View Specifications::Standards::Roadmap
Contains the diagrams that document the Standards Roadmap Viewpoint.

2.8.3.1 View Specifications::Standards::Roadmap::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 2.63 - Standards Roadmap

Elements

- ActualEnterprisePhase
- Forecast
- Protocol
- Standard
- SubjectOfForecast

2.8.4 View Specifications::Standards::Traceability

Contains the diagrams that document the Standards Traceability Viewpoint.

2.8.4.1 View Specifications::Standards::Traceability::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 2.64 - Standards Traceability
2.9 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.

2.9.1 View Specifications::Actual Resources::Taxonomy

Contains the diagrams that document the Actual Resources Taxonomy Viewpoint.

2.9.1.1 View Specifications::Actual Resources::Taxonomy::Actual Resources Taxonomy

Stakeholders: Solution Providers, Systems Engineers, Human Resources, Business Architects
Concerns: the hierarchy of control within actual organizations, between actual posts and actual persons filling those actual posts that affect the architecture and how it is used. It is the instance version of the personnel structure which defines the types of organizations and post, etc.
Definition: illustrates the actual organizational structure and relationships among actual organizations, actual posts, and actual persons filling those actual post, that are the key players in the architecture.
Recommended Implementation: SysML Block Definition Diagram, SysML Internal Block Diagram

Figure 2.65 - Actual Resources Taxonomy
Elements

- ActualMeasurement
- ActualOrganization
- ActualOrganizationalResource
- ActualPerson
- ActualPost
- ActualPropertySet
- ActualResource
- ActualResponsibility
- ActualResponsibleResource
- CapabilityConfiguration
- FieldedCapability
- Organization
- Person
- Post
- ResourcePerformer
- Responsibility

2.9.2 View Specifications::Actual Resources::Structure

Contains the diagrams that document the Actual Resources Structure Viewpoint.

2.9.2.1 View Specifications::Actual Resources::Structure::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.
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 2.66 - Actual Resources Structure

Elements

- ActualOrganization
- ActualOrganizationalResource
- ActualPerson
- ActualPost

Unified Architecture Framework (UAF), v1.0
2.9.3 View Specifications::Actual Resources::Connectivity

Contains the diagrams that document the Actual Resources Connectivity Viewpoint.
2.9.3.1 View Specifications::Actual Resources::Connectivity::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

---

Elements

- ActualOrganization
- ActualOrganizationalResource
- ActualPerson

---

Figure 2.67 - Actual Resources Connectivity
**View Specifications::Actual Resources::Traceability**

Contains the diagrams that document the Actual Resources Traceability Viewpoint.

**View Specifications::Actual Resources::Traceability::Actual Resources Traceability**

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 8:70 - Actual Resources Traceability**

**Elements**
- ActualResource
- Capability
- CapableElement
- Exhibits

---

### 2.10 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:
- elements and relationships that are involved in defining the environments applicable to capability, operational concept, or set of systems.
- measurable properties that can be used to support analysis such as KPIs, MoEs, TPIs, etc.
2.10.1 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 2.68 - Dictionary
Elements

- Alias
- Definition
- Information
- SameAs
- UAFEElement

2.11 View Specifications::Summary & 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.
2.11.1 View Specifications::Summary & 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 2.69 - Summary & Overview

Elements

- ActualEnterprisePhase
- ActualOrganizationalResource
- ArchitecturalDescription
- ArchitecturalReference
- Architecture
- ArchitectureMetadata
2.12 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.

2.12.1 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 2.70 - Requirements
Elements

- Refine
- Requirement
- Satisfy
- Trace
- UAFEElement
- Verify

2.13  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

2.13.1  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 2.71 - Information Model

Elements

- DataElement
- DataModel
- DataRole
- Implements
- InformationElement
- InformationRole
- OperationalAsset
- ResourceAsset
- DataElement
- DataModel
- Implements
- InformationElement
2.14 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.

2.14.1 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 2.72 - Parameters: Environment
Elements

- Activity
  - ActivityPerformableUnderCondition
- ActualCondition
- ActualEnvironment
- ActualLocation
- ActualPropertySet
- ActualResource
2.14.2 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, MoEs, 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 2.73 - Parameters: Measurements
2.15 View Specifications::Other
Contains the diagrams that document the use of BPMN, NIEM, IEPPV in the context of UAF.

2.15.1 View Specifications::Other::BPMN
Stakeholders: Business Architects, 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 using BPMN.
Recommended Implementation: BPMN Process Diagram
Figure 2.74 - BPMN Elements
• AssetRole
• BPMN2Metamodel::BPMNMessage
• BPMN2Metamodel::CallActivity
• BPMN2Metamodel::MessageFlow
• BPMN2Metamodel::Process
• BPMN2Metamodel::ResourceRole
• BPMN2Metamodel::SequenceFlow
• Exchange
• ExchangeItem
• InteractionMessage
• InteractionRole
• InteractionScenario
• Process
• ProcessEdge
• ProcessUsage
• InformationElement
• IsCapableToPerform
• OperationalActivity
• OperationalActivityAction
• OperationalActivityEdge
• OperationalAgent
• OperationalExchange
• OperationalExchangeItem
• OperationalPerformer
• OperationalRole
• PerformsInContext
• ResourcePerformer

2.15.2 View Specifications::Other::IEPPV

Stakeholders: Data Modelers, Solution Providers, Systems Engineers, Software Engineers, Systems Architects, Business Architects, information architects
Concerns: information exchanges, information interfaces, information interoperability, information sharing and safeguarding
Definition: UAFP supports information modeling and traceability to IEPPV model elements using the IEPPV-defined elements: Message, SemanticElement, and FilteredSemanticElement, used to represent data, properties/attributes, structure, format, and relationships. The IEPPV profile enables the specification of the policies, rules and constraints governing the packaging (assembly, transformation, marking, redaction) of data elements conforming to information sharing and safeguarding requirements. The IEPPV profile also governs the processing (parsing, transformation, and marshalling) received information and data element.
Recommended Implementation: UML Class Diagram, SysML Block Diagram
2.15.3 View Specifications::Other::NIEM

Stakeholders: Data Modelers, Solution Providers, Systems Engineers, Software Engineers, Systems Architects, Business Architects

Concerns: information exchanges, information interoperability, data schema

Definition: A specification representing the structure, semantics, and relationships of data objects that satisfy an information exchange requirement. Used for organizing and packaging Model Package Descriptions (MPDs) and Information Exchange Package Documentation (IEPD) as defined by the National Information Exchange Model (NIEM). An IEPD is a type of MPD. The NIEM MPD defines an Enterprise Information Exchange Model (EIEM) as an MPD that contains NIEM-conforming schemas that define and declare data components to be consistently reused in the IEPDs of an enterprise. An EIEM is a collection of schemas organized into a collection of subset schemas and one or more extension schemas. An information sharing enterprise creates and maintains an EIEM.

Recommended Implementation: UML Class Diagram, SysML Block Diagram
Figure 2.76 - NIEM

Elements

- Abstraction
- DataElement
- DataModel
- DataObject
- InformationElement
- InformationModel
0 93  Domain Metamodel (DMM) Elements

93.1 Domain MetaModel

This package contains the elements of the DMM.

8.1.3 Domain MetaModel::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.

**Domain MetaModel::Metadata::Taxonomy**

**ArchitectureMetadata**

**Package:** Taxonomy

**isAbstract:** No

**Generalization:** Metadata

**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 9:1 - ArchitectureMetadata](image)

**InteractionScenarioGeneralization**

**Package:** Taxonomy

**isAbstract:** No

**Generalization:** UML2.5Metamodel::Generalization, MeasurableElement

**Description**

A InteractionScenarioGeneralization is a taxonomic relationship between a more general InteractionScenario and a more specific InteractionScenario.
**Metadata**

**Package:** Taxonomy  
**isAbstract:** No  
**Generalization:** MeasurableElement

**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.

**Attributes**
- **category:** String [0..1]  
- **dublinCoreTag:** String [0..1]  
- **metaDataScheme:** String [0..1]  
- **name:** String [0..1]

**ProcessGeneralization**

**Package:** Taxonomy  
**isAbstract:** No  
**Generalization:** UML2.5Metamodel::Generalization, MeasurableElement

**Description**
A ProcessGeneralization is a taxonomic relationship between a more general Process and a more specific Process.
**PropertySetGeneralization**

**Package:** Taxonomy

**isAbstract:** No

**Generalization:** UML2.5Metamodel::Generalization, MeasurableElement

**Description**
A PropertySetGeneralization is a taxonomic relationship between a more general PropertySet and a more specific PropertySet.

---

**StateDescriptionGeneralization**

**Package:** Taxonomy

**isAbstract:** No

**Generalization:** UML2.5Metamodel::Generalization, MeasurableElement

**Description**
A StateDescriptionGeneralization is a taxonomic relationship between a more general StateDescription and a more specific StateDescription.
**Domain MetaModel::Metadata::Structure**

**EnvironmentProperty**

- **Package:** Structure
- **isAbstract:** No
- **Generalization:** MeasurableElement

**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.

**Domain MetaModel::Metadata::Connectivity**

**Exchange**

- **Package:** Connectivity
- **isAbstract:** Yes
- **Generalization:** MeasurableElement, BPMN2Metamodel::MessageFlow, SubjectOfSecurityConstraint

**Description**

Abstract tuple, grouping OperationalExchanges and ResourceExchanges that exchange Resources.
**Exchangeltem**

**Package:** Connectivity  
**isAbstract:** Yes  
**Generalization:** BPMN2Metamodel::BPMNMessage

**Description**
An abstract grouping for elements that defines the types of elements that can be exchanged between Assets and conveyed by an Exchange.

**Figure 9:9 - Exchangeltem**

**Resource**

**Package:** Connectivity  
**isAbstract:** Yes  
**Generalization:** PropertySet

**Description**
Abstract type grouping all elements that can be conveyed by an Exchange.
Domain MetaModel::Metadata::Processes

**ActivityPerformableUnderCondition**

*Package: Processes*

*isAbstract: No*

*Generalization: MeasurableElement*

**Description**

The ActualCondition under which an Activity is performed.

**IsCapableToPerform**

*Package: Processes*

*isAbstract: No*

*Generalization: MeasurableElement*

**Description**

A tuple defining the traceability between the structural elements to the Activities that they can perform.
PerformsInContext

Package: Processes
isAbstract: No
Generalization: MeasurableElement

Description
A tuple 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.
**ProcessEdge**

**Package:** Processes

**isAbstract:** Yes

**Generalization:** MeasurableElement, UML2.5Metamodel::Activity, UML2.5Metamodel::ActivityEdge, BPMN2Metamodel::SequenceFlow

**Description**

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

---

**Figure 9:16 - ProcessEdge**

Unified Architecture Framework (UAF), v1.0
**ProcessOperation**

**Package:** Processes

**isAbstract:** Yes

**Generalization:** MeasurableElement, UML2.5Metamodel::Activity, UML2.5Metamodel::Operation

**Description**

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

![Diagram of ProcessOperation](image)

---

**ProcessParameter**

**Package:** Processes

**isAbstract:** Yes

**Generalization:** MeasurableElement, UML2.5Metamodel::Activity, UML2.5Metamodel::CallBehaviorAction, UML2.5Metamodel::Parameter

**Description**

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

![Diagram of ProcessParameter](image)

---

**ProcessUsage**

**Package:** Processes

**isAbstract:** Yes

**Generalization:** MeasurableElement, UML2.5Metamodel::Activity, UML2.5Metamodel::CallBehaviorAction, BPMN2Metamodel::CallActivity

**Description**

An abstract type that represents a behavior or process (i.e. a Function or OperationalActivity) that can be performed by a Performer.
**Domain MetaModel::Metadata::States**

**StateDescription**

*Package:* States  
*isAbstract:* Yes  
*Generalization:* UML2.5Metamodel::StateMachine

**Description**

An abstract type that represents a state machine (i.e. an OperationalStateDescription or ResourceStateDescription), depicting how the Asset responds to various events and the actions.

**Domain MetaModel::Metadata::Interaction Scenarios**

**InteractionMessage**

*Package:* Interaction Scenarios  
*isAbstract:* Yes  
*Generalization:* MeasurableElement, UML2.5Metamodel::Activity, BPMN2Metamodel::Process, UML2.5Metamodel::Interaction, UML2.5Metamodel::Message

**Description**

An abstract type that groups several types of messages used in the InteractionScenario.

**InteractionRole**

*Package:* Interaction Scenarios
**isAbstract:** Yes

**Generalization:** BPMN2Metamodel::ResourceRole

**Description**
An abstract type that represents an individual participant in the InteractionScenario.

---

**InteractionScenario**

**Package:** Interaction Scenarios

**isAbstract:** Yes

**Generalization:** MeasurableElement, UML2.5Metamodel::Activity, BPMN2Metamodel::Process, UML2.5Metamodel::Interaction

**Description**
An abstract type that specifies interactions between Assets, like ResourcePerformers, and ServiceSpecifications.

---

**Domain MetaModel::Metadata::Information**

**Information**

**Package:** Information

**isAbstract:** No

**Generalization:** MeasurableElement

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

**Domain MetaModel::Metadata::Constraints**

**Rule**

**Package:** Constraints  
**isAbstract:** Yes  
**Generalization:** MeasurableElement

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

**Domain MetaModel::Metadata::Traceability**

**ArchitecturalReference**

**Package:** Traceability
**isAbstract:** No  
**Generalization:** MeasurableElement

**Description**  
A tuple that specifies that one architectural description refers to another.

![ArchitecturalReference Diagram](image-url)

**Figure 9:25 - ArchitecturalReference**

**Implements**  
**Package:** Traceability  
**isAbstract:** No  
**Generalization:** MeasurableElement

**Description**  
A tuple that defines how an element in the upper layer of abstraction is implemented by a semantically equivalent element (for example tracing the Functions to the OperationalActivities) in the lower level of abstraction.

![Implements Diagram](image-url)

**Figure 9:26 - Implements**
8.1.4 Domain MetaModel::Strategic

Domain MetaModel::Strategic::Taxonomy

Capability
Package: Taxonomy
isAbstract: No
Generalization: PropertySet, Desirer

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

Figure 9:27 - Capability

CapabilityGeneralization
Package: Taxonomy
isAbstract: No
Generalization: PropertySetGeneralization

Description
A CapabilityGeneralization is a taxonomic relationship between a more general Capability and a more specific Capability.

Figure 9:28 - CapabilityGeneralization
Domain MetaModel::Strategic::Structure

ActualEnduringTask

Package: Structure
isAbstract: No
Generalization: CapableElement, ActualPropertySet

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.

![Diagram of ActualEnduringTask]

Figure 9:29 - ActualEnduringTask

ActualEnterprisePhase

Package: Structure
isAbstract: No
Generalization: CapableElement, ActualPropertySet, Achiever

Description
An individual that describes the phase of an actual enterprise endeavor.

![Diagram of ActualEnterprisePhase]

Figure 9:30 - ActualEnterprisePhase

CapabilityRole

Package: Structure
isAbstract: No
Generalization: PropertySet, Desirer, MeasurableElement

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

-
**CapabilityRole**

**EnduringTask**

**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.

**EnterprisePhase**

**Package:** Structure  
**isAbstract:** No  
**Generalization:** PropertySet

**Description**  
A type of a current or future state of the enterprise.

![Diagram of EnterprisePhase](image)

**Figure 9:34 - EnterprisePhase**

**EnterpriseVision**

**Package:** Structure  
**isAbstract:** No  
**Generalization:** PropertySet

**Description**  
A Vision describes the future state of the enterprise, without regard to how it is to be achieved. BMM: OMG dtc-13-08-24.

![Diagram of EnterpriseVision](image)

**Figure 9:35 - EnterpriseVision**

**StructuralPart**

**Package:** Structure  
**isAbstract:** No  
**Generalization:** MeasurableElement

**Description**  
A current or future state of the wholeLifeEnterprise or another EnterprisePhase.
**TemporalPart**

**Package:** Structure  
**isAbstract:** No  
**Generalization:** MeasurableElement

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

**VisionStatement**

**Package:** Structure  
**isAbstract:** No  
**Generalization:** MeasurableElement

**Description**  
A type of comment that describes the future state of the enterprise, without regard to how it is to be achieved. BMM: OMG dtc-13-08-24.

**WholeLifeEnterprise**

**Package:** Structure  
**isAbstract:** No  
**Generalization:** EnterprisePhase
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 9:39 - WholeLifeEnterprise

Domain MetaModel::Strategic::Connectivity

CapabilityDependency
Package: Connectivity
isAbstract: No
Generalization: MeasurableElement

Description
A tuple that asserts that one CapabilityDependency is dependent from another.

Figure 9:40 - CapabilityDependency

CapabilityRoleDependency
Package: Connectivity
isAbstract: No
Generalization: MeasurableElement

Figure 9:41 - CapabilityRoleDependency
Domain MetaModel::Strategic::States

**AchievedEffect**

**Package:** States  
**isAbstract:** No  
**Generalization:** MeasurableElement

**Description**

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

**Figure 9:42 - AchievedEffect**

**Achiever**

**Package:** States  
**isAbstract:** Yes  
**Generalization:** UAFElement

**Description**

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

**Figure 9:43 - Achiever**

**DesiredEffect**

**Package:** States  
**isAbstract:** No  
**Generalization:** MeasurableElement

**Description**

A tuple relating the Desirer (a Capability or OrganizationalResource) to an ActualState.
**Figure 9:44 - DesiredEffect**

**Desirer**

**Package:** States  
**isAbstract:** Yes  
**Generalization:** UAFElement

**Description**  
Abstract type used to group architecture elements that might desire a particular effect.

**Figure 9:45 - Desirer**

**Domain MetaModel::Strategic::Traceability**

**CapabilityForTask**

**Package:** Traceability  
**isAbstract:** No  
**Generalization:** MeasurableElement

**Description**  
A tuple that asserts that a Capability is required in order for an Enterprise to conduct a phase of an EnduringTask.
Figure 9:46 - CapabilityForTask

**CapabilityElement**

*Package: Traceability*

*isAbstract: Yes*

*Generalization: UAFElement*

**Description**

An abstract type that represents a structural element that can exhibit capabilities.

Figure 9:47 - CapableElement

**Exhibits**

*Package: Traceability*

*isAbstract: No*

*Generalization: MeasurableElement*

**Description**

A tuple that exists between a CapableElement and a Capability that it meets under specific environmental conditions.
Figure 9:48 - Exhibits

MapsToCapability
Package: Traceability
isAbstract: No
Generalization: MeasurableElement

Description
A tuple denoting that an Activity contributes to providing a Capability.

Figure 9:49 - MapsToCapability

OrganizationInEnterprise
Package: Traceability
isAbstract: No
Generalization: MeasurableElement

Description
A tuple relating an ActualOrganization to an ActualEnterprisePhase to denote that the ActualOrganization plays a role or is a stakeholder in an ActualEnterprisePhase.

Figure 9:50 - OrganizationInEnterprise
8.1.5 Domain MetaModel::Operational

Domain MetaModel::Operational::Taxonomy

**ArbitraryConnector**

**Package:** Taxonomy

**isAbstract:** No

**Generalization:** MeasurableElement

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

![ArbitraryConnector Diagram](image1)

**ConceptItem**

**Package:** Taxonomy

**isAbstract:** Yes

**Generalization:** UAFEElement

**Description**
Abstract, an item which may feature in a HighLevelOperationalConcept.

![ConceptItem Diagram](image2)

**HighLevelOperationalConcept**

**Package:** Taxonomy

**isAbstract:** No

![HighLevelOperationalConcept Diagram](image3)
**Generalization:** PropertySet

**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.

![Diagram of PropertySet](image)

**Figure 9:53 - HighLevelOperationalConcept**

**Domain MetaModel::Operational::Structure**

**KnownResource**
**Package:** Structure
**isAbstract:** No
**Generalization:** OperationalPerformer, ResourcePerformer

**Description**
Asserts that a known ResourcePerformer constrains the implementation of the OperationalPerformer that plays the role in the OperationalArchitecture.

![Diagram of KnownResource](image)

**Figure 9:54 - KnownResource**

**OperationalAgent**
**Package:** Structure
**isAbstract:** Yes
**Generalization:** SubjectOfOperationalConstraint, CapableElement, OperationalAsset, Desirer

**Description**
An abstract type grouping OperationalArchitecture and OperationalPerformer.
**OperationalAgent**

**Package:** Structure  
**isAbstract:** No  
**Generalization:** OperationalAgent, Architecture

**Description**  
A type used to denote a model of the Architecture, described from the Operational perspective.

**OperationalArchitecture**

**Package:** Structure  
**isAbstract:** No  
**Generalization:** OperationalAgent, Architecture

**Description**  
A type used to denote a model of the Architecture, described from the Operational perspective.

**OperationalMethod**

**Package:** Structure  
**isAbstract:** No  
**Generalization:** ProcessOperation

**Description**  
A behavioral feature of an OperationalAgent whose behavior is specified in an OperationalActivity.
**OperationalMethod**

**OperationalParameter**

*Package*: Structure  
*isAbstract*: No  
*Generalization*: ProcessParameter

**Description**

A type that represents inputs and outputs of an OperationalActivity. It is typed by an OperationalExchangeItem.

---

**OperationalPerformer**

*Package*: Structure  
*isAbstract*: No  
*Generalization*: OperationalAgent

**Description**

A logical entity that IsCapableToPerform OperationalActivities which produce, consume and process Resources.
**Figure 9:59 - OperationalPerformer**

**OperationalRole**

**Package:** Structure  
**isAbstract:** No  
**Generalization:** LocationHolder, AssetRole, InteractionRole

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

**Figure 9:60 - OperationalRole**

**ProblemDomain**

**Package:** Structure  
**isAbstract:** No  
**Generalization:** OperationalRole

**Description**  
A property associated with an OperationalArchitecture, used to specify the scope of the problem.
Figure 9:61 - ProblemDomain

Domain MetaModel::Operational::Connectivity

OperationalConnector

Package: Connectivity
isAbstract: No
Generalization: MeasurableElement

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

Figure 9:62 - OperationalConnector

OperationalExchange

Package: Connectivity
isAbstract: No
Generalization: Exchange, SubjectOfOperationalConstraint

Description
Asserts that a flow can exist between OperationalPerformers (i.e. flows of information, people, materiel, or energy).
**Figure 9:63 - OperationalExchange**

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

**OperationalExchangeItem**
- **Package**: Connectivity
- **isAbstract**: Yes
- **Generalization**: Resource, SubjectOfSecurityConstraint, ExchangeItem

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

---

**Figure 9:64 - OperationalExchangeItem**
**OperationalInterface**

**Package:** Connectivity  
**isAbstract:** No  
**Generalization:** PropertySet  

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

![Figure 9:65 - OperationalInterface](image)

**OperationalPort**

**Package:** Connectivity  
**isAbstract:** No  
**Generalization:** MeasurableElement  

**Description**  
An interaction point for an OperationalAgent through which it can interact with the outside environment and which is defined by an OperationalInterface.

![Figure 9:66 - OperationalPort](image)

**OperationalSignal**

**Package:** Connectivity  
**isAbstract:** No  
**Generalization:** SubjectOfOperationalConstraint, OperationalExchangeItem  

**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).
**Domain MetaModel::Operational::Processes**

**OperationalActivity**

**Package:** Processes  
**isAbstract:** No  
**Generalization:** SubjectOfOperationalConstraint, Process  

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

**OperationalActivityAction**

**Package:** Processes  
**isAbstract:** No  
**Generalization:** ProcessUsage  

**Description**  
*A call of an OperationalActivity in the context of another OperationalActivity.*
Figure 9:69 - OperationalActivityAction

**OperationalActivityEdge**

- **Package:** Processes
- **isAbstract:** Yes
- **Generalization:** ProcessEdge

**Description**

A tuple that shows the flow of Resources (objects/information) between OperationalActivityActions.

Figure 9:70 - OperationalActivityEdge

**StandardOperationalActivity**

- **Package:** Processes
- **isAbstract:** No
- **Generalization:** OperationalActivity

**Description**

A sub-type of OperationalActivity that is a standard operating procedure.
Domain MetaModel::Operational::States

OperationalStateDescription

Package: States
isAbstract: No
Generalization: MeasurableElement, StateDescription

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

Domain MetaModel::Operational::Interaction Scenarios

OperationalInteractionScenario

Package: Interaction Scenarios
isAbstract: No
Generalization: InteractionScenario

Description
A specification of the interactions between OperationalPerformers in an OperationalArchitecture.
Figure 9:73 - OperationalInteractionScenario

**OperationalMessage**

**Package:** Interaction Scenarios  
**isAbstract:** No  
**Generalization:** InteractionMessage

**Description**  
Message for use in an OperationalInteractionScenario which carries any of the subtypes of OperationalExchange.

Figure 9:74 - OperationalMessage

**Domain MetaModel::Operational::Information**

**InformationElement**

**Package:** Information  
**isAbstract:** No  
**Generalization:** SubjectOfOperationalConstraint, OperationalAsset, OperationalExchangeItem

**Description**  
An item of information that flows between OperationalPerformers and is produced and consumed by the OperationalActivities that the OperationalPerformers are capable to perform (see IsCapableToPerform).
**Domain MetaModel::Operational::Constraints**

**OperationalConstraint**

**Package:** Constraints  
**isAbstract:** No  
**Generalization:** Rule

**Description**

A Rule governing an operational architecture element i.e. OperationalPerformer, OperationalActivity, InformationElement etc.

**SubjectOfOperationalConstraint**

**Package:** Constraints  
**isAbstract:** Yes  
**Generalization:** UAFElement

**Description**

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

---

**Figure 9:75 - InformationElement**

**Figure 9:76 - OperationalConstraint**

**Figure 9:77 - SubjectOfOperationalConstraint**
8.1.6 Domain MetaModel::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.

**Domain MetaModel::Services::Taxonomy**

**ServiceSpecification**

**Package:** Taxonomy

**isAbstract:** No

**Generalization:** PropertySet, VersionedElement, CapableElement

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

**ServiceSpecificationGeneralization**

**Package:** Taxonomy

**isAbstract:** No

**Generalization:** PropertySetGeneralization

**Description**
A ServiceSpecificationGeneralization is a taxonomic relationship between a more general ServiceSpecification and a more specific ServiceSpecification.
**Domain MetaModel::Services::Structure**

**ServiceConnector**

**Package:** Structure  
**isAbstract:** No  
**Generalization:** MeasurableElement

**Description**

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

**ServiceMethod**

**Package:** Structure  
**isAbstract:** No  
**Generalization:** ProcessOperation

**Description**

A behavioral feature of a ServiceSpecification whose behavior is specified in a ServiceFunction.
Figure 9:81 - ServiceMethod

ServiceParameter

Package: Structure
isAbstract: No
Generalization: ProcessParameter

Description
A type that represents inputs and outputs of a ServiceFunction, represents inputs and outputs of a ServiceSpecification.

Figure 9:82 - ServiceParameter

ServiceSpecificationRole

Package: Structure
isAbstract: No
Generalization: MeasurableElement, InteractionRole

Description
A behavioral feature of a ServiceSpecification whose behavior is specified in a ServiceFunction.
**Domain MetaModel::Services::Connectivity**

**ServiceInterface**

*Package:* Connectivity  
*isAbstract:* No  
*Generalization:* PropertySet

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

**ServicePort**

*Package:* Connectivity  
*isAbstract:* No  
*Generalization:* MeasurableElement

**Description**
An interaction point for a ServiceSpecification through which it can interact with the outside environment and which is defined by a ServiceInterface.
Figure 9:85 - ServicePort

**Domain MetaModel::Services::Processes**

**ServiceFunction**

*Package:* Processes  
*isAbstract:* No  
*Generalization:* Process

**Description**

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

---

Figure 9:86 - ServiceFunction

**ServiceFunctionAction**

*Package:* Processes  
*isAbstract:* No  
*Generalization:* ProcessUsage

**Description**

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

**Package:** Processes

**isAbstract:** Yes

**Generalization:** ProcessEdge

**Description**
A tuple that shows the flow of Resources (objects/information) between OperationalActivityActions.

**ServiceFunctionEdge**

**Domain MetaModel:** Services::States

**ServiceStateDescription**

**Package:** States

**isAbstract:** No

**Generalization:** MeasurableElement, StateDescription

**Description**
A state machine describing the behavior of a ServiceSpecification, depicting how the ServiceSpecification responds to various events and the actions.
Domain MetaModel::Services::Interaction Scenarios

ServiceInteractionScenario

**Package:** Interaction Scenarios

**isAbstract:** No

**Generalization:** InteractionScenario

**Description**

A specification of the interactions between ServiceSpecifications.

ServiceMessage

**Package:** Interaction Scenarios

**isAbstract:** No

**Generalization:** InteractionMessage

**Description**

Message for use in a Service Event-Trace.
**Domain MetaModel::Services::Constraints**

**ServicePolicy**

Package: Constraints  
isAbstract: No  
Generalization: Rule

Description  
A constraint governing the use of one or more ServiceSpecifications.

**Domain MetaModel::Services::Traceability**

**Consumes**

Package: Traceability  
isAbstract: No  
Generalization: MeasurableElement

Description  
A tuple that asserts that an OperationalActivity make use of a service.
8.1.7 Domain MetaModel::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).

**Domain MetaModel::Personnel::Taxonomy**

**Organization**

**Package:** Taxonomy

**isAbstract:** No

**Generalization:** OrganizationalResource

**Description**

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

**OrganizationalResource**
**Person**

**Package:** Taxonomy  
**isAbstract:** No  
**Generalization:** OrganizationalResource

**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).

**Post**

**Package:** Taxonomy  
**isAbstract:** No  
**Generalization:** OrganizationalResource

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

**Package:** Taxonomy  
**isAbstract:** No  
**Generalization:** OrganizationalResource

**Description**  
The type of duty required of a Person or Organization.

**Domain MetaModel::Personnel::Structure**

**PostRole**

**Package:** Structure  
**isAbstract:** No  
**Generalization:** OrganizationalResource, ResourceRole

**Description**  
A usage of a post in the context of another OrganizationalResource. Creates a whole-part relationship.
**SubOrganization**

*Package:* Structure

*isAbstract:* No

*Generalization:* OrganizationalResource, ResourceRole

**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).

---

**Domain MetaModel::Personnel::Connectivity**

**Command**

*Package:* Connectivity

*isAbstract:* No

*Generalization:* ResourceExchange

**Description**

A type of ResourceExchange that asserts that one OrganizationalResource commands another.
Unified Architecture Framework (UAF), v1.0

**ResourceExchange**

**Control**

**Package:** Connectivity

**isAbstract:** No

**Generalization:** ResourceExchange

**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).

**Domain MetaModel::Personnel::Interaction Scenarios**

**ResourceInteractionScenario**

**Package:** Interaction Scenarios

**isAbstract:** No

**Generalization:** InteractionScenario

**Description**

A specification of the interactions between ResourcePerformers in a ResourceArchitecture.
**Domain MetaModel::Personnel::Constraints**

**Competence**

*Package:* Constraints  
*isAbstract:* No  
*Generalization:* PropertySet, SubjectOfForecast

**Description**

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

**CompetenceForRole**

*Package:* Constraints  
*isAbstract:* No  
*Generalization:* MeasurableElement

**Description**

A tuple used to associate an organizational role with a specific set of required competencies.
**RequiresCompetence**

**Package:** Constraints  
**isAbstract:** No  
**Generalization:** MeasurableElement

**Description**  
A tuple that asserts that an ActualOrganizationalResource is required to have a specific set of Competencies.

![Diagram of RequiresCompetence](image)

**Figure 9:106 - RequiresCompetence**

**Domain MetaModel::Personnel::Roadmap**

**FillsPost**  
**Package:** Roadmap  
**isAbstract:** No  
**Generalization:** MeasurableElement

**Description**  
A tuple that asserts that an ActualPerson fills an ActualPost.

![Diagram of FillsPost](image)

**Figure 9:107 - 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.

**Domain MetaModel::Personnel::Traceability**

**CompetenceToConduct**

**Package:** Traceability  
**isAbstract:** No  
**Generalization:** MeasurableElement
Description
A tuple used to associate a Function with a specific set of Competencies needed to conduct the Function.

Figure 9:108 - CompetenceToConduct

8.1.8 Domain MetaModel::Resources

Domain MetaModel::Resources::Taxonomy

CapabilityConfiguration
Package: Taxonomy
isAbstract: No
Generalization: ResourceArchitecture

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

Figure 9:109 - CapabilityConfiguration

NaturalResource
Package: Taxonomy
isAbstract: No
Generalization: PhysicalResource

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

Figure 9:110 - NaturalResource

PhysicalResource
Package: Taxonomy
isAbstract: Yes

**Generalization:** ResourcePerformer

**Description**
An abstract type defining physical resources (i.e. OrganizationalResource, ResourceArtifact and NaturalResource).

**Figure 9:111 - PhysicalResource**

**ResourceArchitecture**

**Package:** Taxonomy

**isAbstract:** No

**Generalization:** ResourcePerformer, Architecture

**Description**
A type used to denote a model of the Architecture, described from the ResourcePerformer perspective.

**Figure 9:112 - ResourceArchitecture**

**ResourceArtifact**

**Package:** Taxonomy

**isAbstract:** No

**Generalization:** PhysicalResource

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

**Package:** Taxonomy

**isAbstract:** Yes.

**Generalization:** ResourceExchangeItem, SubjectOfResourceConstraint, OperationalExchangeItem, SubjectOfForecast, CapableElement, Desirer, VersionedElement, ResourceAsset

**Description**

An abstract grouping of elements that can perform Functions.

**Attributes**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>isStandardConfiguration</td>
<td>Indicates if the ResourcePerformer is StandardConfiguration, default=false.</td>
</tr>
</tbody>
</table>
**Software**

Package: Taxonomy

isAbstract: No

Generalization: ResourceArtifact

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

![Diagram of Software](image1)

**System**

Package: Taxonomy

isAbstract: No

Generalization: ResourceArchitecture

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).

![Diagram of System](image2)

**Domain MetaModel::Resources::Structure**

**ResourceMethod**

Package: Structure

isAbstract: No

Generalization: ProcessOperation

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

**Package**: Structure

**isAbstract**: No

**Generalization**: ProcessParameter

**Description**

A type that represents inputs and outputs of a Function. It is typed by a ResourceInteractionItem.

---

**ResourceParameter**

**Package**: Structure

**isAbstract**: No

**Generalization**: ProcessParameter

**Description**

A type that represents inputs and outputs of a Function. It is typed by a ResourceInteractionItem.

---

**ResourcePort**

**Package**: Structure

**isAbstract**: No

**Generalization**: ProtocolImplementation, MeasurableElement

**Description**

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

**Package:** Structure

**isAbstract:** No

**Generalization:** SubjectOfResourceConstraint, LocationHolder, AssetRole, InteractionRole

**Description**

**Figure 9:120 - ResourceRole**

**Domain MetaModel::Resources::Connectivity**

**ResourceConnector**

- **Package:** Connectivity
- **isAbstract:** No
- **Generalization:** ProtocolImplementation, MeasurableElement

**Description**

A channel for exchange between two ResourceRoles.
Figure 9:121 - ResourceConnector

**ResourceExchange**

**Package:** Connectivity  
**isAbstract:** No  
**Generalization:** Exchange

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

Figure 9:122 - ResourceExchange

**ResourceExchangeItem**

**Package:** Connectivity  
**isAbstract:** Yes  
**Generalization:** Resource, SubjectOfSecurityConstraint, ExchangeItem

**Description**  
Resource, SubjectOfSecurityConstraint, ExchangeItem
An abstract type grouping elements that defines the types of elements that can be exchanged between ResourcePerformers and conveyed by a ResourceExchange.

**Figure 9:123 - ResourceExchangeItem**

**ResourceInterface**

**Package:** Connectivity

**isAbstract:** No

**Generalization:** PropertySet

**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.
### ResourceInterface

**Package:** Connectivity  
**isAbstract:** No  
**Generalization:** ResourceExchangeItem

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

### ResourceSignal

**Package:** Connectivity  
**isAbstract:** No  
**Generalization:** ResourceExchangeItem

**Description**  
An Activity which is specified in the context to the ResourcePerformer (human or machine) that IsCapableToPerform it.

### Domain MetaModel::Resources::Processes

**Function**

**Package:** Processes  
**isAbstract:** No  
**Generalization:** SubjectOfResourceConstraint, Process

**Description**  
An Activity which is specified in the context to the ResourcePerformer (human or machine) that IsCapableToPerform it.
**Figure 9:126 - Function**

**FunctionAction**

**Package:** Processes  
**isAbstract:** No  
**Generalization:** ProcessUsage

**Description**

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

---

**Figure 9:127 - FunctionAction**

**FunctionEdge**

**Package:** Processes  
**isAbstract:** No  
**Generalization:** ProcessEdge

**Description**

A tuple that shows the flow of Resources (objects/data) between FunctionActions.
Figure 9:128 - FunctionEdge

**Domain MetaModel::Resources::States**

**ResourceStateDescription**

**Package:** States

**isAbstract:** No

**Generalization:** MeasurableElement, StateDescription

**Description**

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

Figure 9:129 - ResourceStateDescription

**Domain MetaModel::Resources::Interaction Scenarios**

**ResourceMessage**

**Package:** Interaction Scenarios

**isAbstract:** No

**Generalization:** InteractionMessage

**Description**

Message for use in a Resource Event-Trace which carries any of the subtypes of ResourceExchange.
**Domain MetaModel::Resources::Information**

**DataElement**

- **Package:** Information
- **isAbstract:** No
- **Generalization:** SubjectOfResourceConstraint, ResourceAsset, ResourceExchangeItem

**Description**

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

**DataRole**

- **Package:** Information
- **isAbstract:** No
- **Generalization:** AssetRole

**Description**
A usage of DataElement that exists in the context of an ResourceAsset. It also allows the representation of the whole-part aggregation of DataElements.

**Domain MetaModel::Resources::Constraints**

**ResourceConstraint**

- **Package:** Constraints
- **isAbstract:** No
- **Generalization:** Rule

**Description**

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

**SubjectOfResourceConstraint**

- **Package:** Constraints
- **isAbstract:** Yes
- **Generalization:** UAFElement

**Description**

An abstract type grouping elements that can be the subject of a ResourceConstraint.
**Domain MetaModel::Resources::Roadmap**

**Forecast**
*Package:* Roadmap  
*isAbstract:* No  
*Generalization:* MeasurableElement

**Description**
A tuple 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.

**SubjectOfForecast**
*Package:* Roadmap  
*isAbstract:* Yes  
*Generalization:* UAFElement

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

**Technology**
*Package:* Roadmap  
*isAbstract:* No  
*Generalization:* ResourceArtifact

**Description**
A sub type of ResourceArtifact that indicates a technology domain, i.e. nuclear, mechanical, electronic, mobile telephony etc.
Figure 9:137 - Technology

**VersionedElement**

**Package:** Roadmap  
**isAbstract:** Yes  
**Generalization:** UAFElement

**Description**  
An abstract type grouping ResourcePerformer and ServiceSpecification that allows VersionOfConfiguration to be related to ActualProjectMilestones.

Figure 9:138 - VersionedElement

**VersionOfConfiguration**

**Package:** Roadmap  
**isAbstract:** No  
**Generalization:** MeasurableElement

**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 9:139 - VersionOfConfiguration**

**VersionSuccession**

*Package:* Roadmap  
*isAbstract:* No  
*Generalization:* MeasurableElement

**Description**

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

**Figure 9:140 - VersionSuccession**

**WholeLifeConfiguration**

*Package:* Roadmap  
*isAbstract:* No  
*Generalization:* PropertySet

**Description**

A set of VersionedElements.
**Domain MetaModel::Resources::Traceability**

**ProtocolImplementation**

*Package:* Traceability

*isAbstract:* Yes

*Generalization:* UAFElement

**Description**

An abstract type grouping architectural elements that can implement Protocols.

---

**8.1.9 Domain MetaModel::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.

**Domain MetaModel::Security::Taxonomy**

**Asset**

*Package:* Taxonomy

*isAbstract:* Yes

*Generalization:* SubjectOfForecast, ConceptItem, LocationHolder, PropertySet, SubjectOfSecurityConstraint

**Description**

Asset as applied to Security views, an abstract type that indicates the types of elements that can be considered as a subject for security analysis.
Figure 9:143 - Asset

**OperationalAsset**

**Package:** Taxonomy

**isAbstract:** Yes

**Generalization:** Asset

Figure 9:144 - OperationalAsset

**OperationalMitigation**

**Package:** Taxonomy

**isAbstract:** No

**Generalization:** OperationalArchitecture

**Description**

A set of OperationalPerformers intended to address against specific operational risks.
Figure 9:145 - OperationalMitigation

**ResourceAsset**
*Package: Taxonomy*
*isAbstract: Yes*
*Generalization: Asset*

**Figure 9:146 - ResourceAsset**

**ResourceMitigation**
*Package: Taxonomy*
*isAbstract: No*
*Generalization: ResourceArchitecture*

Description
A set of ResourcePerformers intended to address against specific risks.

**Figure 9:147 - ResourceMitigation**

**SecurityEnclave**
*Package: Taxonomy*
*isAbstract: No*
*Generalization: ResourceArchitecture*

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.
Domain MetaModel::Security::Structure

**AssetRole**

* Package: Structure
* isAbstract: Yes
* Generalization: BPMN2Metamodel::ResourceRole, SubjectOfSecurityConstraint, MeasurableElement

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.

**InformationRole**

* Package: Structure
* isAbstract: No
* Generalization: AssetRole

Description

A usage of InformationElement that exists in the context of an OperationalAsset. It also allows the representation of the whole-part aggregation of InformationElements.
**Domain MetaModel::Security::Processes**

**EnhancedSecurityControl**

- **Package:** Processes
- **isAbstract:** No
- **Generalization:** SecurityControl

**Description**

Statement of security capability to: (i) build in additional but related, functionality to a basic control; and/or (ii) increase the strength of a basic control.

**Enhances**

- **Package:** Processes
- **isAbstract:** No
- **Generalization:** MeasurableElement

**Description**

A tuple relating the EnhancedSecurityControl to a SecurityControl.
**Protects**

**Package:** Processes  
**isAbstract:** No  
**Generalization:** MeasurableElement

**Description**
A tuple that asserts that a SecurityControl is required to protect an Asset.

![Diagram](Figure 9:153 - Protects)

**ProtectsInContext**

**Package:** Processes  
**isAbstract:** No  
**Generalization:** MeasurableElement

**Description**
A tuple 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.

![Diagram](Figure 9:154 - ProtectsInContext)

**SecurityProcess**

**Package:** Processes  
**isAbstract:** No  
**Generalization:** OperationalActivity, Function, SubjectOfSecurityConstraint

**Description**
The security-related procedure that satisfies the security control requirement.
**SecurityProcessAction**

**Package:** Processes  
**isAbstract:** No  
**Generalization:** OperationalActivityAction, FunctionAction  

**Description**  

---

**Domain MetaModel::Security::Constraints**

**ActualRisk**

**Package:** Constraints  
**isAbstract:** No  
**Generalization:** ActualPropertySet  

**Description**  
**Caveat**

**Package:** Constraints  
**isAbstract:** No  
**Generalization:** SecurityConstraint

**Description**  
A statement that details alternate conditions under which the rule is not valid.

---

**Risk**

**Package:** Constraints  
**isAbstract:** No  
**Generalization:** PropertySet

**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.
SecurityAvailability

**Package:** Constraints
**isAbstract:** No
**Generalization:** SecurityMeasurement

**Description**
Details the potential impact on organization or individuals if the information is not available to those who need to access it.

SecurityCategory

**Package:** Constraints
**isAbstract:** No
**Generalization:** MeasurementSet

**Description**
The security categories that have been determined for each type of information processed, stored, or transmitted by those information systems. The generalized format for expressing the security category (SC) of an information system is:

SC information system = \{(confidentiality, impact), (integrity, impact), (availability, impact)\}.
**SecurityClassification**

*Package: Constraints*

*isAbstract: No*

*Generalization: SecurityMeasurement*

**Description**

Details a classification for the exchange.

**SecurityClassificationKind**

*Package: Constraints*

*isAbstract: No*

*Generalization: MeasurableElement*

**Description**

A type that defines acceptable values for the security category (SC) of an information system, where the acceptable values for potential impact are low, moderate, or high.

**SecurityConstraint**

*Package: Constraints*

*isAbstract: No*

*Generalization: Rule*

**Description**

A type of rule that captures a formal statement to define access control policy language.
Figure 9:164 - SecurityConstraint

SecurityControl

Package: Constraints
isAbstract: No
Generalization: MeasurableElement

Description
The management, operational, and technical control (i.e., safeguard or countermeasure) prescribed for an information system to protect the confidentiality, integrity, and availability of the system and its information [NIST SP 800-53].

Figure 9:165 - SecurityControl

SecurityControlFamily

Package: Constraints
isAbstract: No

Unified Architecture Framework (UAF), v1.0
**Generalization:** SecurityControl

**Description**
An element that organizes security controls into a family. Each Security Control Family contains security controls related to the general security topic of the family.

**SecurityIntegrity**

**Package:** Constraints

**isAbstract:** No

**Generalization:** SecurityMeasurement

**Description**
Details the potential impact on organization or individuals due to modification or destruction of information, and includes ensuring information non-repudiation and authenticity.

**SecurityMeasurement**

**Package:** Constraints

**isAbstract:** Yes

**Generalization:** Measurement

**Description**
An abstract type grouping all types of security measurements (e.g. SecurityIntegrity, SecurityAvailability).
**SubjectOfSecurityConstraint**

**Package:** Constraints

**isAbstract:** Yes

**Generalization:** UAFElem

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

**Domain MetaModel::Security::Traceability**

**Affects**

**Package:** Traceability

**isAbstract:** No

**Generalization:** MeasurableElement

**Description**
A tuple that asserts that a Risk is applicable to an Asset.
**AffectsInContext**

**Description**
A tuple that asserts that a Risk is applicable to an AssetRole in the specific context or configuration.

**Mitigates**

**Description**
A tuple relating a Security Control to a Risk. Mitigation is established to manage risk and could be represented as an overall strategy or through techniques (mitigation configurations) and procedures (SecurityProcesses).

**OwnsRisk**

**Description**
A tuple relating a Risk to an organizational resource that is responsible for executing the risk mitigation.
**OwnsRiskInContext**

**Package:** Traceability  
**isAbstract:** No  
**Generalization:** MeasurableElement

**Description**  
A tuple relating a Risk to an organizational role that is responsible for executing the risk mitigation in the specific context or configuration.

---

**Figure 9:174 - OwnsRiskInContext**

---

**8.1.10 Domain MetaModel::Projects**

**Domain MetaModel::Projects::Taxonomy**

**Project**

**Package:** Taxonomy  
**isAbstract:** No  
**Generalization:** OrganizationalResource

**Description**  
A type that describes types of time-limited endeavors that are required to meet one or more Capability needs.

---

**Figure 9:175 - Project**

---

**ProjectMilestone**

**Package:** Taxonomy  
**isAbstract:** No  
**Generalization:** PropertySet

---

Unified Architecture Framework (UAF), v1.0
Description
A type of event in a Project by which progress is measured.

Domain MetaModel::Projects::Structure

ActualProjectMilestoneRole
Package: Structure
isAbstract: No
Generalization: ActualState
Description
An ActualProjectMilestone that is applied to a ProjectMilestoneRole.

ProjectMilestoneRole
Package: Structure
isAbstract: No
Generalization: MeasurableElement
Description
The role played by a ProjectMilestone in the context of a Project.
**ProjectMilestoneRole**

**Package:** Structure  
**isAbstract:** No  
**Generalization:** ActualState

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

**ProjectStatus**

**Package:** Structure  
**isAbstract:** No  
**Generalization:** ActualState

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

**ProjectTheme**

**Package:** Structure  
**isAbstract:** No  
**Generalization:** MeasurableElement

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

**Package:** Structure  
**isAbstract:** No  
**Generalization:** MeasurableElement

**Description**  
An enumerated type that specifies a status for a ProjectTheme.

---

**Domain MetaModel::Projects::Connectivity**

**MilestoneDependency**

**Package:** Connectivity  
**isAbstract:** No  
**Generalization:** MeasurableElement

**Description**  
A tuple between two ActualProjectMilestones that denotes one ActualProjectMilestone follows from another.
**Domain MetaModel::Projects::Processes**

**ProjectActivity**
- **Package:** Processes
- **isAbstract:** No
- **Generalization:** Function, Process

**Description**
An activity carried out during a project.

**ProjectActivityAction**
- **Package:** Processes
- **isAbstract:** No
- **Generalization:** FunctionAction

**Description**
The ProjectActivityAction is defined as a call behavior action that invokes the activity that needs to be performed.
Domain MetaModel::Projects::Interaction Scenarios

ProjectSequence
Package: Interaction Scenarios
isAbstract: No
Generalization: MeasurableElement

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

Figure 9:185 - ProjectSequence

Domain MetaModel::Projects::Roadmap

ActualProject
Package: Roadmap
isAbstract: No
Generalization: ActualOrganizationalResource, Achiever

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

Figure 9:186 - ActualProject

ActualProjectMilestone
Package: Roadmap
isAbstract: No
Generalization: ActualPropertySet

Description

Figure 9:186 - ActualProjectMilestone

Unified Architecture Framework (UAF), v1.0
An event with a start date in an ActualProject from which progress is measured.

**Figure 9:187 - ActualProjectMilestone**

**Constraints**

[1] unnamed1 startTime=endTime

**Domain MetaModel::Projects::Traceability**

**ResponsibleFor**

**Package:** Traceability  
**isAbstract:** No  
**Generalization:** MeasurableElement

**Description**

A tuple between an ActualResponsibleResource and an ActualResponsibility or ActualProject. It defines the duties that the ActualResponsibleResource is ResponsibleFor.

**Figure 9:188 - ResponsibleFor**

**8.1.11 Domain MetaModel::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.

**Domain MetaModel::Standards::Taxonomy**

**Protocol**
- **Package:** Taxonomy
- **isAbstract:** No
- **Generalization:** Standard

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

![Diagram of Protocol and ProtocolStack](image)

**ProtocolStack**
- **Package:** Taxonomy
- **isAbstract:** No
- **Generalization:** Protocol

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

![Diagram of ProtocolStack](image)

**Standard**
- **Package:** Taxonomy
- **isAbstract:** No
- **Generalization:** SubjectOfForecast, PropertySet
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 9:191 - 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.

Domain MetaModel::Standards::Structure

ProtocolLayer
Package: Structure
isAbstract: No
Generalization: MeasurableElement

Description

8.1.12 Domain MetaModel::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.
Domain MetaModel::Actual Resources::Taxonomy

ActualOrganization

Package: Taxonomy  
IsAbstract: No  
Generalization: Actual Responsible Resource

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

ActualOrganizationalResource

Package: Taxonomy  
IsAbstract: Yes  
Generalization: Actual Resource, Stakeholder

Description
Abstract element for an Actual Organization, Actual Person or Actual Post.

Figure 9:193 - 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.

Figure 9:194 - ActualOrganizationalResource
**ActualPerson**

Package: Taxonomy

isAbstract: No

Generalization: ActualResponsibleResource

Description
An individual human being.

**Figure 9:195 - ActualPerson**

**ActualPost**

Package: Taxonomy

isAbstract: No

Generalization: ActualResponsibleResource

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 9:196 - ActualPost**

**ActualResource**

Package: Taxonomy

isAbstract: No

Generalization: ActualPropertySet, SubjectOfResourceConstraint, Achiever, CapableElement

Description
An individual, fully-realized ResourcePerformer.
**ActualResource**

**ActualResourceRelationship**

**Package:** Taxonomy

**isAbstract:** No

**Generalization:** UAFElement

**Description**

An actual resource flow existing between ActualResources (i.e. flow of data, people, materiel, or energy).

**ActualResponsibility**

**Package:** Taxonomy

**isAbstract:** No

**Generalization:** ActualOrganizationalResource

**Description**

An actual duty required of a Person or Organization.
**ActualResponsibleResource**

**Package:** Taxonomy  
**isAbstract:** Yes  
**Generalization:** ActualOrganizationalResource

**Description**
An abstract type grouping responsible OrganizationalResources.

**FieldedCapability**

**Package:** Taxonomy  
**isAbstract:** No  
**Generalization:** ActualResource

**Description**
An individual, fully-realized capability.
Domain MetaModel::Actual Resources::Constraints

**ActualService**
- Package: Constraints
- isAbstract: Yes
- Generalization: ActualMeasurementSet, CapableElement

Description
An individual ServiceSpecification.

![ActualService Diagram]

**ProvidedServiceLevel**
- Package: Constraints
- isAbstract: No
- Generalization: ActualService

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

![ProvisionedServiceLevel Diagram]

**ProvidesCompetence**
- Package: Constraints
- isAbstract: No
- Generalization: MeasurableElement

Description
A tuple that asserts that an ActualOrganizationalResource provides a specific set of Competencies.
Figure 9:204 - ProvidesCompetence

**RequiredServiceLevel**

*Package:* Constraints  
*isAbstract:* No  
*Generalization:* ActualService

**Description**

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

Figure 9:205 - RequiredServiceLevel

### 8.1.13 Domain MetaModel::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.

**Recommended Implementation:** Tabular format, SysML Block Definition Diagram.

**Alias**

*Package:* Dictionary  
*isAbstract:* No  
*Generalization:* MeasurableElement

**Description**

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

Attributes

- **nameOwner**: String[*]

  Someone or something that uses this alternative name.

**Definition**

Package: Dictionary

*isAbstract*: No

*Generalization*: MeasurableElement

Description

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

**SameAs**

Package: Dictionary

*isAbstract*: No

*Generalization*: MeasurableElement

Description

A tuple that asserts that two elements refer to the same real-world thing.
8.1.14 Domain MetaModel::Summary & Overview

ArchitecturalDescription

Package: Summary & Overview
isAbstract: No
Generalization: MeasurableElement

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.

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.
methodologyUsed : String[*]  The methodology used in developing the architecture.
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.

status : String[*]  Approval status of the architecture.
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.

version : String[*]  Version number of the architecture.

Architecture

Package: Summary & Overview
isAbstract: Yes
Generalization: UAFElement

Description
An abstract type that represents a generic architecture. Subtypes are OperationalArchitecture and PhysicalArchitecture.

Concern

Package: Summary & Overview
isAbstract: No
Generalization: PropertySet

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

**Package:** Summary & Overview  
**isAbstract:** Yes  
**Generalization:** UAFElement

**Description**  

UAFElement

**Package:** Summary & Overview  
**isAbstract:** Yes

**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 9:213 - UAFElement

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

View
Package: Summary & Overview
isAbstract: No
Generalization: PropertySet

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 9:214 - View

Viewpoint
Package: Summary & Overview

Unified Architecture Framework (UAF), v1.0
**isAbstract**: No  
**Generalization**: PropertySet

**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 9:215 - Viewpoint**

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

**8.1.15 Domain MetaModel::Information**

**DataModel**
- **Package**: Information
- **isAbstract**: No  
  **Generalization**: SubjectOfOperationalConstraint, SubjectOfResourceConstraint

**Description**
A structural specification of data types, showing relationships between them. The type of data captured in the DataModel is described using the enumeration DataModelKind (Conceptual, Logical and Physical).

**Figure 9:216 - DataModel**

**8.1.16 Domain MetaModel::Parameters**

**ActualCondition**
- **Package**: Parameters
**ActualCondition**

*Abstract:* No  
*Generalization:* ActualPropertySet

**Description**  
An individual describing an actual situation with respect to circumstances under which an Operational Activity, Function or Service Function can be performed.

![Diagram of ActualCondition](image)

**ActualEnvironment**

*Abstract:* No  
*Generalization:* ActualCondition

**Description**  
An individual that describes the circumstances of an Environment.

![Diagram of ActualEnvironment](image)

**ActualLocation**

*Abstract:* No  
*Generalization:* ActualCondition

**Description**  
An individual that describes a physical location, for example using text to provide an address, Geo-coordinates, etc.
Figure 9:219 - 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[]  Boolean that indicates if the ActualLocation address is embedded in the ActualLocation name. By default = false.

ActualMeasurement
Package: Parameters
isAbstract: No
Generalization: ActualState

Description
An actual value that is applied to a Measurement.

Figure 9:220 - ActualMeasurement

ActualMeasurementSet
Package: Parameters
isAbstract: No
Generalization: ActualPropertySet

ActualCondition
Description
A set of ActualMeasurements.

**ActualPropertySet**

- **Package:** Parameters
- **isAbstract:** No
- **Generalization:** ActualState

Description
A set or collection of Actual properties.

**ActualState**

- **Package:** Parameters
- **isAbstract:** Yes
- **Generalization:** UAFElement

Description
Abstract element that applies temporal extent to a set of elements realized as Instance Specifications.
Figure 9:223 - ActualState

Attributes

- **endDate** : ISO8601DateTime[0..1]  
  End time for all individual elements.
- **startDate** : ISO8601DateTime[0..1]  
  Start time for all individual elements.

**Condition**

**Package:** Parameters  
**isAbstract:** No  
**Generalization:** PropertySet

**Description**

A type that defines the Location, Environment and/or GeoPoliticalExtent.

Figure 9:224 - Condition

**Environment**

**Package:** Parameters  
**isAbstract:** No  
**Generalization:** Condition

**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 9:225 - Environment

**GeoPoliticalExtentType**

**Package:** Parameters

**isAbstract:** No

**Generalization:** Condition, OperationalExchangeItem, ResourceExchangeItem

**Description**

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

**Figure 9:226 - GeoPoliticalExtentType**

**Attributes**

- `customKind : String` | Captures the kind of GeopoliticalExtentType.

**ISO8601DateTime**

**Package:** Parameters

**isAbstract:** No

**Generalization:** UAFElement

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

**ISO8601DateTime**

### Location

**Package:** Parameters  
**isAbstract:** No  
**Generalization:** ConceptItem, Condition

**Description**

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

**Attributes**

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

### LocationHolder

**Package:** Parameters  
**isAbstract:** Yes  
**Generalization:** UAFEElement

**Description**

Abstract type, used to group elements that are allowed to be associated with a Location.
**Figure 9:229 - LocationHolder**

**MeasurableElement**

**Package:** Parameters  
**isAbstract:** Yes  
**Generalization:** UAFEElement

**Description**  
Abstract type, grouping elements that can be measured by applying MeasurementSets to them.
**Figure 9:230 - MeasurableElement**

**Measurement**

Package: Parameters

isAbstract: No

Generalization: MeasurableElement

Description

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

*Package:* Parameters  
*isAbstract:* No  
*Generalization:* PropertySet

**Description**

A collection of Measurements.

---

**PropertySet**

*Package:* Parameters  
*isAbstract:* Yes  
*Generalization:* UAFElement

**Description**

An abstract type grouping architectural elements that can own Measurements.
3.1.1 Domain MetaModel::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.

3.1.1.1 Domain MetaModel::Metadata::Taxonomy

ArchitectureMetadata

Package: Taxonomy
isAbstract: No
Generalization: MeasurableElement

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.

Metadata

Package: Taxonomy
isAbstract: No
Generalization: MeasurableElement

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.

Attributes
category : String[0..1] Defines the category of a Metadata element example: http://purl.org/dc/terms/abstract. dublinCoreTag :
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.
3.1.1.2 Domain MetaModel::Metadata::Structure

EnvironmentProperty

Package: Structure
isAbstract: No
Generalization: MeasurableElement

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.

3.1.1.3 Domain MetaModel::Metadata::Connectivity

Exchange

Package: Connectivity
isAbstract: Yes
Generalization: MeasurableElement

Description
Abstract tuple, grouping OperationalExchanges and ResourceExchanges that exchange Resources.

Resource

Package: Connectivity
isAbstract: Yes
Generalization: PropertySet

Description
Abstract type grouping all elements that can be conveyed by an Exchange.

3.1.1.4 Domain MetaModel::Metadata::Processes

Activity

Package: Processes
isAbstract: Yes
Generalization: MeasurableElement

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

ActivityPerformableUnderCondition

Package: Processes
isAbstract: No
Generalization: MeasurableElement
Description
The environment under which an Activity is performed.

IsCapableToPerform

Package: Processes
isAbstract: No
Generalization: MeasurableElement

Description
A tuple defining the traceability between the CapableElements to the Activities that they can perform.

PerformsInContext

Package: Processes
isAbstract: No
Generalization: MeasurableElement

Description
A tuple 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.

3.1.1.5 Domain MetaModel::Metadata::Information

Information

Package: Information
isAbstract: No
Generalization: MeasurableElement

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

3.1.1.6 Domain MetaModel::Metadata::Constraints

Rule

Package: Constraints
isAbstract: Yes
Generalization: MeasurableElement

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

3.1.1.7 Domain MetaModel::Metadata::Traceability

ArchitecturalReference

Package: Traceability
isAbstract: No
Generalization: MeasurableElement

Description
A tuple that specifies that one architectural description refers to another.

Implements
Package: Traceability
isAbstract: No
Generalization: MeasurableElement

Description
A tuple that defines how an element in the upper layer of abstraction is implemented by a semantically equivalent element (i.e., tracing the Operational Activities to the Functions that implement them) in the lower level of abstraction.

3.1.2 Domain MetaModel::Strategic

3.1.2.1 Domain MetaModel::Strategic::Taxonomy

Capability
Package: Taxonomy
isAbstract: No
Generalization: PropertySet, Desirer

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

3.1.2.2 Domain MetaModel::Strategic::Structure

ActualEnduringTask
Package: Structure
isAbstract: No
Generalization: CapableElement, ActualPropertySet

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.

ActualEnterprisePhase
Package: Structure
isAbstract: No
Generalization: CapableElement, ActualPropertySet

Description
The ActualState that describes the phase of an Enterprise endeavor.
**CapabilityProperty**

**Package:** Structure

**isAbstract:** No

**Generalization:** MeasurableElement

**Description**

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

**EnduringTask**

**Package:** Structure

**isAbstract:** No

**Generalization:** PropertySet

**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.

**EnterpriseGoal**

**Package:** Structure

**isAbstract:** No

**Generalization:** PropertySet

**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. BMM: OMG dtc-13-08-24.

**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.

**EnterprisePhase**

**Package:** Structure

**isAbstract:** No

**Generalization:** PropertySet

**Description**

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

**EnterpriseVision**

**Package:** Structure

**isAbstract:** No

**Generalization:** UAFElement
Description

A Vision describes the future state of the enterprise, without regard to how it is to be achieved. BMM: OMG dtc-13-08-24.

Attributes


VisionStatement

Package: Structure
isAbstract: No
Generalization: UAFElement

Description

A type of comment that describes the future state of the enterprise, without regard to how it is to be achieved. BMM: OMG dtc-13-08-24.

WholeLifeEnterprise

Package: Structure
isAbstract: No
Generalization: EnterprisePhase

Description

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

3.1.2.3 Domain MetaModel::Strategic::States

AchievedEffect

Package: States
isAbstract: No
Generalization: MeasurableElement

Description

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

Achiever

Package: States
isAbstract: Yes
Generalization: UAFElement

Description

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

*Package: States*

*isAbstract: No*

*Generalization: MeasurableElement*

*Description*

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

**Desirer**

*Package: States*

*isAbstract: Yes*

*Generalization: UAFElement*

*Description*

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

### 3.1.2.4 Domain MetaModel::Strategic::Traceability

**CapabilityForTask**

*Package: Traceability*

*isAbstract: No*

*Generalization: MeasurableElement*

*Description*

A tuple that asserts that a Capability is required in order for an Enterprise to conduct a phase of an EnduringTask.

**CapableElement**

*Package: Traceability*

*isAbstract: Yes*

*Generalization: UAFElement*

*Description*

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

**Exhibits**

*Package: Traceability*

*isAbstract: No*

*Generalization: MeasurableElement*

*Description*

A tuple that exists between a CapableElement and a Capability that it meets under specific environmental conditions.

**MapsToCapability**

*Package: Traceability*
isAbstract: No
Generalization: MeasurableElement

Description
A tuple denoting that an Activity contributes to providing a Capability.

OrganizationInEnterprise

Package: Traceability
isAbstract: No
Generalization: MeasurableElement

Description
A tuple relating an ActualOrganization to an ActualEnterprisePhase to denote that the ActualOrganization plays a role or is a stakeholder in an ActualEnterprisePhase.

3.1.3 Domain MetaModel::Operational

3.1.3.1 Domain MetaModel::Operational::Taxonomy

ConceptItem

Package: Taxonomy
isAbstract: Yes
Generalization: UAFElement

Description
Abstract, an item which may feature in a HighLevelOperationalConcept.

HighLevelOperationalConcept

Package: Taxonomy
isAbstract: No
Generalization: PropertySet

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.

3.1.3.2 Domain MetaModel::Operational::Structure

KnownResource

Package: Structure
isAbstract: No
Generalization: OperationalPerformer

Description
Asserts that a known ResourcePerformer plays a part in the LogicalArchitecture.
**OperationalAgent**

*Package: Structure*

*isAbstract: Yes*

*Generalization: SubjectOfOperationalConstraint, CapableElement, Asset, Desirer*

*Description*

An abstract type grouping LogicalArchitecture and OperationalPerformer.

**OperationalArchitecture**

*Package: Structure*

*isAbstract: No*

*Generalization: OperationalAgent, Architecture*

*Description*

A type used to denote a model of the Architecture, described from the Operational perspective.

**OperationalMethod**

*Package: Structure*

*isAbstract: No*

*Generalization: MeasurableElement*

*Description*

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

**OperationalParameter**

*Package: Structure*

*isAbstract: No*

*Generalization: MeasurableElement*

*Description*

A type that represents inputs and outputs of an OperationalActivity. It is typed by an OperationalExchangeItem.

**OperationalPerformer**

*Package: Structure*

*isAbstract: No*

*Generalization: OperationalAgent*

*Description*

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

**OperationalRole**

*Package: Structure*

*isAbstract: No*
Generalization: LocationHolder, MeasurableElement, AssetRole

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

ProblemDomain

Package: Structure
isAbstract: No
Generalization: OperationalRole

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

3.1.3.3 Domain MetaModel::Operational::Connectivity

OperationalConnector

Package: Connectivity
isAbstract: No
Generalization: MeasurableElement

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

OperationalExchange

Package: Connectivity
isAbstract: No
Generalization: Exchange, SubjectOfOperationalConstraint

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

Attributes

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

OperationalExchangeItem

Package: Connectivity
isAbstract: Yes
Generalization: Resource, SubjectOfSecurityConstraint

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

**Package:** Connectivity

**isAbstract:** No

**Generalization:** PropertySet

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

**OperationalPort**

**Package:** Connectivity

**isAbstract:** No

**Generalization:** MeasurableElement

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

**3.1.3.4 Domain MetaModel::Operational::Processes**

**OperationalActivity**

**Package:** Processes

**isAbstract:** No

**Generalization:** SubjectOfOperationalConstraint::Activity

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

**OperationalActivityAction**

**Package:** Processes

**isAbstract:** No

**Generalization:** MeasurableElement

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

**OperationalActivityEdge**

**Package:** Processes

**isAbstract:** Yes

**Generalization:** MeasurableElement

**Description**
A tuple that shows the flow of Resources (objects/information) between OperationalActivityActions.
**StandardOperationalActivity**

Package: Processes  
isAbstract: No  
Generalization: OperationalActivity  

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

---

**OperationalStateDescription**

Package: States  
isAbstract: No  
Generalization: MeasurableElement  

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

---

**OperationalMessage**

Package: Interaction Scenarios  
isAbstract: No  
Generalization: MeasurableElement  

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

---

**InformationElement**

Package: Information  
isAbstract: No  
Generalization: SubjectOfOperationalConstraint, Asset, OperationalExchangeItem  

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).

---

**OperationalConstraint**

Package: Constraints  
isAbstract: No
Generalization: Rule

Description

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

SubjectOfOperationalConstraint

Package: Constraints
isAbstract: Yes
Generalization: UAFElement

Description

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

3.1.4 Domain MetaModel::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.

3.1.4.1 Domain MetaModel::Services::Taxonomy

ServiceSpecification

Package: Taxonomy
isAbstract: No
Generalization: PropertySet, VersionedElement

Description

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

3.1.4.2 Domain MetaModel::Services::Structure

ServiceConnector

Package: Structure
isAbstract: No
Generalization: MeasurableElement

Description

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

ServiceMethod

Package: Structure
isAbstract: No
Generalization: MeasurableElement
Description
A behavioral feature of a ServiceSpecification whose behavior is specified in a ServiceFunction.

ServiceParameter
Package: Structure
isAbstract: No
Generalization: MeasurableElement
Description
A type that represents inputs and outputs of a ServiceFunction, represents inputs and outputs of a ServiceSpecification.

3.1.4.3 Domain MetaModel::Services::Connectivity
ServiceInterface
Package: Connectivity
isAbstract: No
Generalization: PropertySet
Description
A contract that defines the ServiceMethods and ServiceMessageHandlers that the ServiceSpecification realizes.

ServicePort
Package: Connectivity
isAbstract: No
Generalization: MeasurableElement
Description
An interaction point for a ServiceSpecification through which it can interact with the outside environment and which is defined by a ServiceInterface.

3.1.4.4 Domain MetaModel::Services::Processes
ServiceFunction
Package: Processes
isAbstract: No
Generalization: Activity
Description
An Activity that describes the abstract behavior of ServiceSpecifications, regardless of the actual implementation.

ServiceFunctionAction
Package: Processes
isAbstract: No
Generalization: MeasurableElement
Description

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

3.1.4.5  Domain MetaModel::Services::States

**ServiceStateDescription**

Package: States

isAbstract: No

Generalization: MeasurableElement

Description

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

3.1.4.6  Domain MetaModel::Services::Interaction Scenarios

**ServiceMessage**

Package: Interaction Scenarios

isAbstract: No

Generalization: MeasurableElement

Description

Message for use in a Service Event-Trace.

3.1.4.7  Domain MetaModel::Services::Constraints

**ServicePolicy**

Package: Constraints

isAbstract: No

Generalization: Rule

Description

A constraint governing the use of one or more ServiceSpecifications.

3.1.4.8  Domain MetaModel::Services::Traceability

**Consumes**

Package: Traceability

isAbstract: No

Generalization: MeasurableElement

Description

A tuple that asserts that a service in someway contributes or assists in the execution of an OperationalActivity.
3.1.5 Domain MetaModel::Personnel

Stakeholders: Human resources, Solution Providers, PMs.

Concern: 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).

3.1.5.1 Domain MetaModel::Personnel::Taxonomy

Organization

Package: Taxonomy

isAbstract: No

Generalization: OrganizationalResource

Description

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

OrganizationalResource

Package: Taxonomy

isAbstract: Yes

Generalization: PhysicalResource, Stakeholder

Description

An abstract type for Organization, Person, Post and Responsibility.

Person

Package: Taxonomy

isAbstract: No

Generalization: OrganizationalResource

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.).

Post

Package: Taxonomy

isAbstract: No

Generalization: OrganizationalResource

Description

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

Responsibility

Package: Taxonomy
**Abstract:**

The type of duty required of a Person or Organization.

### 3.1.5.2 Domain MetaModel::Personnel::Connectivity

**Command**

*Package: Connectivity*

**Description**

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

**Control**

*Package: Connectivity*

**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).

### 3.1.5.3 Domain MetaModel::Personnel::Constraints

**Competence**

*Package: Constraints*

**Description**

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

**CompetenceForRole**

*Package: Constraints*

**Description**

A tuple used to associate an organizational role with a specific set of required competencies.

**RequiresCompetence**

*Package: Constraints*
Abstract: No
Generalization: MeasurableElement
Description
A tuple that asserts that an ActualOrganizationalResource is required to have a specific set of Competencies.

3.1.5.4 Domain MetaModel::Personnel::Traceability

CompetenceToConduct
Package: Traceability
isAbstract: No
Generalization: MeasurableElement
Description
A tuple used to associate a Function with a specific set of Competencies needed to conduct the Function.

3.1.6 Domain MetaModel::Resources

3.1.6.1 Domain MetaModel::Resources::Taxonomy

CapabilityConfiguration
Package: Taxonomy
isAbstract: No
Generalization: ResourceArchitecture
Description
A composite structure representing the physical and human resources (and their interactions) in an enterprise, assembled to meet a capability.

NaturalResource
Package: Taxonomy
isAbstract: No
Generalization: PhysicalResource
Description
Type of physical resource that occurs in nature such as oil, water, gas, or coal.

PhysicalResource
Package: Taxonomy
isAbstract: Yes
Generalization: ResourcePerformer
Description
An abstract type defining physical resources (i.e., OrganizationalResource, ResourceArtifact and NaturalResource).
**ResourceArchitecture**

*Package: Taxonomy*

*isAbstract: No*

*Generalization: ResourcePerformer, Architecture*

*Description*

A type used to denote a model of the Architecture, described from the ResourcePerformer perspective.

**ResourceArtifact**

*Package: Taxonomy*

*isAbstract: No*

*Generalization: PhysicalResource*

*Description*

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

**ResourcePerformer**

*Package: Taxonomy*

*isAbstract: Yes*

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

*Description*

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

*Attributes*

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

**Software**

*Package: Taxonomy*

*isAbstract: No*

*Generalization: ResourceArtifact*

*Description*

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

**System**

*Package: Taxonomy*

*isAbstract: No*

*Generalization: ResourceArchitecture*

*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).

3.1.6.2 Domain MetaModel::Resources::Structure

ResourceMethod

Package: Structure

isAbstract: No

Generalization: MeasurableElement

Description

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

ResourceParameter

Package: Structure

isAbstract: No

Generalization: MeasurableElement

Description

A type that represents inputs and outputs of a Function. It is typed by a ResourceInteractionItem.

ResourcePort

Package: Structure

isAbstract: No

Generalization: ProtocolImplementation, MeasurableElement

Description

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

ResourceRole

Package: Structure

isAbstract: No

Generalization: SubjectOfResourceConstraint, LocationHolder, MeasurableElement, AssetRole

Description


3.1.6.3 Domain MetaModel::Resources::Connectivity

ResourceConnector

Package: Connectivity

isAbstract: No

Generalization: ProtocolImplementation, MeasurableElement
Description
A channel for exchange between two ResourceRoles.

ResourceExchange
Package: Connectivity
isAbstract: No
Generalization: Exchange

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

ResourceExchangeItem
Package: Connectivity
isAbstract: Yes
Generalization: Resource, SubjectOfSecurityConstraint

Description
An abstract type grouping elements that define the types of elements that can be exchanged between ResourcePerformers and conveyed by a ResourceExchange.

ResourceInterface
Package: Connectivity
isAbstract: No
Generalization: PropertySet

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.

3.1.6.4 Domain MetaModel::Resources::Processes

Function
Package: Processes
isAbstract: No
Generalization: SubjectOfResourceConstraint, Activity

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

FunctionAction
Package: Processes
isAbstract: No
Generalization: MeasurableElement
Description

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

**FunctionEdge**

**Package:** Processes  
**isAbstract:** No  
**Generalization:** MeasurableElement

Description

A tuple that shows the flow of Resources (objects/data) between FunctionActions.

3.1.6.5 Domain MetaModel::Resources::States

**ResourceStateDescription**

**Package:** States  
**isAbstract:** No  
**Generalization:** MeasurableElement

Description

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

3.1.6.6 Domain MetaModel::Resources::Interaction Scenarios

**ResourceMessage**

**Package:** Interaction Scenarios  
**isAbstract:** No  
**Generalization:** MeasurableElement

Description


3.1.6.7 Domain MetaModel::Resources::Information

**DataElement**

**Package:** Information  
**isAbstract:** No  
**Generalization:** SubjectOfResourceConstraint, SubjectOfSecurityConstraint, Asset, ResourceExchangeItem

Description

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

3.1.6.8 Domain MetaModel::Resources::Constraints

**ResourceConstraint**

**Package:** Constraints

---

114

Unified Architecture Framework (UAF), v1.0
**Abstract:**

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

**SubjectOfResourceConstraint**

**Package:** Constraints

**isAbstract:** Yes

**Generalization:** UAFElement

**Description**

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

**3.1.6.9 Domain MetaModel::Resources::Roadmap**

**Forecast**

**Package:** Roadmap

**isAbstract:** No

**Generalization:** MeasurableElement

**Description**

A tuple 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.

**SubjectOfForecast**

**Package:** Roadmap

**isAbstract:** Yes

**Generalization:** UAFElement

**Description**

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

**Technology**

**Package:** Roadmap

**isAbstract:** No

**Generalization:** ResourceArtifact

**Description**

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

**VersionedElement**

**Package:** Roadmap

**isAbstract:** Yes
Generalization: UAFElement

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

VersionOfConfiguration

Package: Roadmap
isAbstract: No
Generalization: MeasurableElement

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

VersionSuccession

Package: Roadmap
isAbstract: No
Generalization: MeasurableElement

Description
A tuple between two VersionOfConfigurations that denotes that one VersionOfConfiguration follows from another.

WholeLifeConfiguration

Package: Roadmap
isAbstract: No
Generalization: PropertySet

Description
A set of VersionedElements.

3.1.6.10 Domain MetaModel::Resources::Traceability

ProtocolImplementation

Package: Traceability
isAbstract: Yes
Generalization: UAFElement

Description
An abstract type grouping architectural elements that can implement Protocols.

3.1.7 Domain MetaModel::Security


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.

### 3.1.7.1 Domain MetaModel::Security::Taxonomy

#### Asset

**Package**: Taxonomy  
**isAbstract**: Yes  
**Generalization**: SubjectOfForecast, ConceptItem, LocationHolder, PropertySet, SubjectOfSecurityConstraint

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

#### OperationalMitigation

**Package**: Taxonomy  
**isAbstract**: No  
**Generalization**: OperationalArchitecture

**Description**  
A set of measures intended to address against specific operational risks. Comprises of a subset of activities that are performed in mitigation of the risk to protect the asset that is the subject of risk (OperationalRole). In the case of a SecurityRisk, the form of activity is a SecurityControl, otherwise it is an OperationalActivity.

#### ResourceMitigation

**Package**: Taxonomy  
**isAbstract**: No  
**Generalization**: ResourceArchitecture

**Description**  
A set of measures intended to implement an OperationalMitigation. Comprises a subset of activities that are performed in mitigation of the risk to protect the asset that is the subject of risk (ResourceRole) at the physical level. In the case of a Risk applicable to security, the form of activity is a SecurityControl or an EnhancedSecurityControl, otherwise it is a Function.

#### SecurityEnclave

**Package**: Taxonomy  
**isAbstract**: No  
**Generalization**: ResourceArchitecture

**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.
3.1.7.2 Domain MetaModel::Security::Structure

AssetRole

Package: Structure

isAbstract: Yes

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.

SecurityProperty

Package: Structure

isAbstract: No

Generalization: MeasurableElement, AssetRole

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 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.

3.1.7.3 Domain MetaModel::Security::Processes

EnhancedSecurityControl

Package: Processes

isAbstract: No

Generalization: SecurityControl

Description

A type of Activity that represents an enhanced SecurityControl. It 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.

Enhances

Package: Processes

isAbstract: No

Generalization: MeasurableElement

Description

A tuple relating the EnhancedSecurityControl to a SecurityControl.

Protects

Package: Processes
isAbstract: No
Generalization: MeasurableElement

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

ProtectsInContext
Package: Processes
isAbstract: No
Generalization: MeasurableElement

Description
A tuple 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.

SecurityControl
Package: Processes
isAbstract: No
Generalization: OperationalActivity, Function

Description
A type of OperationalActivity that specifies a safeguard or countermeasure prescribed for OperationalPerformer. It is intended to protect the confidentiality, integrity, and availability of its information.

SecurityControlAction
Package: Processes
isAbstract: No
Generalization: OperationalActivityAction, FunctionAction

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 at OperationalPerformer (OperationalRole).

SecurityControlFamily
Package: Processes
isAbstract: No
Generalization: MeasurableElement

Description
A type that organizes security controls into a family.

3.1.7.4 DomainMetaModel::Security::Constraints

ActualRisk
Package: Constraints
Abstract: No

Generalization: ActualPropertySet

Description


Caveat

Package: Constraints

isAbstract: No

Generalization: SecurityConstraint

Description

A statement that details alternate conditions under which the rule is not valid.

Risk

Package: Constraints

isAbstract: No

Generalization: PropertySet

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.

SecurityAvailability

Package: Constraints

isAbstract: No

Generalization: SecurityMeasurement

SecurityCategory

Package: Constraints

isAbstract: No

Generalization: MeasurementSet

SecurityClassification

Package: Constraints

isAbstract: No

Generalization: SecurityMeasurement
Description

The security categories that have been determined for each type of information processed, stored, or transmitted by those information systems. The generalized format for expressing the security category (SC) of an information system is:
SC information system = {(confidentiality, impact), (integrity, impact), (availability, impact)}.

**SecurityClassificationKind**

Package: Constraints

isAbstract: No

Description

A type that defines acceptable values for the security category (SC) of an information system, where the acceptable values for potential impact are low, moderate, or high.

**SecurityConstraint**

Package: Constraints

isAbstract: No

Generalization: Rule

Description

A type of rule that captures a formal statement to define access control policy language.

**SecurityIntegrity**

Package: Constraints

isAbstract: No

Generalization: SecurityMeasurement

**SecurityMeasurement**

Package: Constraints

isAbstract: Yes

Generalization: Measurement

**SubjectOfSecurityConstraint**

Package: Constraints

isAbstract: Yes

Generalization: UAFElement

Description

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

**3.1.7.5 Domain-MetaModel::Security::Traceability**

**Affects**

Package: Traceability

isAbstract: No
Description
A dependency that asserts that a Risk is applicable to an AssetRole.

Mitigates
Package: Traceability
isAbstract: No

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

OwnsRisk
Package: Traceability
isAbstract: No

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

3.1.8 Domain MetaModel::Project

3.1.8.1 Domain MetaModel::Project::Taxonomy

Project
Package: Taxonomy
isAbstract: No
Generalization: Desirer, PropertySet

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

ProjectMilestone
Package: Taxonomy
isAbstract: No
Generalization: PropertySet

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

3.1.8.2 Domain MetaModel::Project::Structure

ActualProjectMilestoneRole
Package: Structure
isAbstract: No
Generalization: ActualState
Description

An ActualProjectMilestone that is applied to a ProjectMilestoneRole.

**ProjectMilestoneRole**

**Package:** Structure  
**isAbstract:** No  
**Generalization:** MeasurableElement

Description

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

**ProjectStatus**

**Package:** Structure  
**isAbstract:** No  
**Generalization:** ActualState

Description

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

**ProjectTheme**

**Package:** Structure  
**isAbstract:** No  
**Generalization:** MeasurableElement

Description

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

**StatusIndicators**

**Package:** Structure  
**isAbstract:** No  
**Generalization:** MeasurableElement

Description

An enumerated type that specifies a status for a ProjectTheme.

### 3.1.8.3 Domain MetaModel::Project::Connectivity

**MilestoneDependency**

**Package:** Connectivity  
**isAbstract:** No  
**Generalization:** MeasurableElement

Description

A tuple between two ActualProjectMilestones that denotes one ActualProjectMilestone follows from another.
3.1.8.4 Domain MetaModel::Project::Interaction Scenarios

ProjectSequence

Package: Interaction Scenarios
isAbstract: No
Generalization: MeasurableElement

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

3.1.8.5 Domain MetaModel::Project::Roadmap

ActualProject

Package: Roadmap
isAbstract: No
Generalization: ActualPropertySet

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

ActualProjectMilestone

Package: Roadmap
isAbstract: No
Generalization: ActualPropertySet

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

Constraints
[1] unnamed1 startTime=endTime

3.1.8.6 Domain MetaModel::Project::Traceability

ResponsibleFor

Package: Traceability
isAbstract: No
Generalization: MeasurableElement

Description
A tuple between an ActualResponsibleResource and an ActualResponsibility or ActualProject. It defines the duties that the ActualResponsibleResource is ResponsibleFor.

3.1.9 Domain MetaModel::Standards


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.

3.1.9.1 Domain MetaModel::Standards::Taxonomy

**Protocol**

*Package: Taxonomy*

*isAbstract:* No

**Generalization:** Standard

**Description**

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

**ProtocolStack**

*Package: Taxonomy*

*isAbstract:* No

**Generalization:** Protocol

**Description**

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

**Standard**

*Package: Taxonomy*

*isAbstract:* No

**Generalization:** SubjectOfForecast, PropertySet

**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.

**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.

3.1.9.2 Domain MetaModel::Standards::Structure

**ProtocolLayer**

*Package: Structure*

*isAbstract:* No

**Generalization:** MeasurableElement

**Description**

3.1.10 Domain MetaModel::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.

3.1.10.1 Domain MetaModel::Actual Resources::Taxonomy

**ActualOrganization**

**Package:** Taxonomy

isAbstract: No

**Generalization:** ActualResponsibleResource

**Description**

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

**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.

**ActualOrganizationalResource**

**Package:** Taxonomy

isAbstract: Yes

**Generalization:** ActualResource, Stakeholder

**Description**

Abstract element for an ActualOrganization, ActualPerson, or ActualPost.

**ActualPerson**

**Package:** Taxonomy

isAbstract: No

**Generalization:** ActualResponsibleResource

**Description**

An individual human being.

**ActualPost**

**Package:** Taxonomy

isAbstract: No

**Generalization:** ActualResponsibleResource
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.

**ActualResource**

**Package:** Taxonomy

**isAbstract:** No

**Generalization:** ActualPropertySet, LocationHolder, SubjectOfResourceConstraint, Achiever

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

**ActualResourceRelationship**

**Package:** Taxonomy

**isAbstract:** No

**Generalization:** UAFElement

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

**ActualResponsibility**

**Package:** Taxonomy

**isAbstract:** No

**Generalization:** ActualOrganizationalResource

**Description**
An actual duty required of a Person or Organization.

**ActualResponsibleResource**

**Package:** Taxonomy

**isAbstract:** Yes

**Generalization:** ActualOrganizationalResource

**Description**
An abstract type grouping responsible OrganizationalResources.

**FieldedCapability**

**Package:** Taxonomy

**isAbstract:** No

**Generalization:** ActualResource

**Description**
An individual, fully-realized capability.
3.1.10.2 Domain MetaModel::Actual Resources::Constraints

**ActualService**

*Package:* Constraints  
*isAbstract:* Yes  
*Generalization:* [ActualMeasurementSet](#)

**Description**

An individual ServiceSpecification.

**ProvidedServiceLevel**

*Package:* Constraints  
*isAbstract:* No  
*Generalization:* [ActualService](#)

**Description**

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

**ProvidesCompetence**

*Package:* Constraints  
*isAbstract:* No  
*Generalization:* [MeasurableElement](#)

**Description**

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

**RequiredServiceLevel**

*Package:* Constraints  
*isAbstract:* No  
*Generalization:* [ActualService](#)

**Description**

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

3.1.11 Domain MetaModel::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.

**Recommended Implementation:** Tabular format, SysML Block Definition Diagram.
**Alias**

*Package: Dictionary*  
*isAbstract: No*  
*Generalization: MeasurableElement*

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

**Attributes**

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

**Definition**

*Package: Dictionary*  
*isAbstract: No*  
*Generalization: MeasurableElement*

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

**Attributes**

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

**SameAs**

*Package: Dictionary*  
*isAbstract: No*  
*Generalization: UAFElement*

**Description**  
A tuple that asserts that two elements refer to the same real-world thing.

### 3.1.12 Domain MetaModel::Summary & Overview

**ArchitecturalDescription**

*Package: Summary & Overview*  
*isAbstract: No*  
*Generalization: MeasurableElement*

**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.
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 is 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 made during development so far. Updates may be made 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 development tools the ArchitecturalDescription uses as well as file names and formats if appropriate.


Architecture

Package: Summary & Overview

isAbstract: Yes

Generalization: UAEElement

Description

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

Concern

Package: Summary & Overview

isAbstract: No

Generalization: PropertySet

Description

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

**Package: Summary & Overview**

**isAbstract:** Yes

**Generalization:** UAFElement

**Description**

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

**UAFElement**

**Package: Summary & Overview**

**isAbstract:** Yes

**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.

**Attributes**


**View**

**Package: Summary & Overview**

**isAbstract:** No

**Generalization:** PropertySet

**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)].

**Viewpoint**

**Package: Summary & Overview**

**isAbstract:** No

**Generalization:** PropertySet

**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)].

**Attributes**

language : String[*] The languages used to express the Viewpoint.

method : String[*] The methods employed in the development of the Viewpoint.

purpose : String[0..1] The purpose of the Viewpoint.
3.1.13 Domain MetaModel::Information

DataModel

Package: Information
isAbstract: No

Generalization: SubjectOfOperationalConstraint, SubjectOfResourceConstraint

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 enumeration DataModelKind (Conceptual, Logical, and Physical).

3.1.14 Domain MetaModel::Parameters

ActualCondition

Package: Parameters
isAbstract: No

Generalization: ActualPropertySet

Description

The ActualState of an environment or location describing its situation.

ActualEnvironment

Package: Parameters
isAbstract: No

Generalization: ActualCondition

Description

The ActualState that describes the circumstances of an Environment.

ActualLocation

Package: Parameters
isAbstract: No

Generalization: ActualCondition

Description

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

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[] Boolean that indicates if the ActualLocation address is embedded in the ActualLocation name. By default = false.
**ActualMeasurement**

Package: Parameters
isAbstract: No
Generalization: ActualState

Description
An actual value that is applied to a Measurement.

**ActualMeasurementSet**

Package: Parameters
isAbstract: No
Generalization: ActualPropertySet

Description
A set of ActualMeasurements.

**ActualPropertySet**

Package: Parameters
isAbstract: No
Generalization: ActualState

Description
A set or collection of Actual properties.

**ActualState**

Package: Parameters
isAbstract: Yes
Generalization: UAFElement

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

Attributes

endDate : ISO8601DateTime[0..1] —— End time for all individual elements.

startDate : ISO8601DateTime[0..1] —— Start time for all individual elements.

**Condition**

Package: Parameters
isAbstract: No
Generalization: PropertySet
Description

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

Environment

Package: Parameters

isAbstract: No

Generalization: Condition

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.

GeoPoliticalExtentType

Package: Parameters

isAbstract: No

Generalization: Condition, OperationalExchangeItem, ResourceExchangeItem

Description

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

Attributes

customKind : String[] —— Captures the kind of GeoPoliticalExtentType.

ISO8601DateTime

Package: Parameters

isAbstract: No

Generalization: UAFElement

Description

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

Location

Package: Parameters

isAbstract: No

Generalization: ConceptItem, Condition

Description

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

Attributes

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

*Package:* Parameters  
*isAbstract:* Yes  
*Generalization:* UAFElement

**Description**

Abstract type, used to group elements that are allowed to be associated with a Location.

**MeasurableElement**

*Package:* Parameters  
*isAbstract:* Yes  
*Generalization:* UAFElement

**Description**

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

**Measurement**

*Package:* Parameters  
*isAbstract:* No  
*Generalization:* MeasurableElement

**Description**

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

**MeasurementSet**

*Package:* Parameters  
*isAbstract:* No  
*Generalization:* PropertySet

**Description**

A collection of Measurements.

**PropertySet**

*Package:* Parameters  
*isAbstract:* Yes  
*Generalization:* UAFElement

**Description**

An abstract type grouping architectural elements that can own Measurements.
This page intentionally left blank.