Architecture Frameworks
Rich Modeling Support for TOGAF, Zachman, UDPM, Archimate
# Table of Contents

Architecture Frameworks ........................................................................................................ 7

TOGAF ...................................................................................................................................... 8

Welcome to TOGAF in Enterprise Architect ........................................................................... 10

TOGAF Copyright Notices ....................................................................................................... 12

TOGAF Software Product License Agreement ......................................................................... 13

Acknowledgement of Trademarks ............................................................................................ 15

TOGAF Support .......................................................................................................................... 16

TOGAF System Requirements .................................................................................................. 17

Getting Started With TOGAF ................................................................................................... 18

Using TOGAF ............................................................................................................................ 19

TOGAF Model Patterns ............................................................................................................. 20

The TOGAF Interface Diagram ............................................................................................... 21

The TOGAF Model Structure ................................................................................................... 23

The TOGAF Diagrams ............................................................................................................... 24

The TOGAF Toolbox Pages ....................................................................................................... 25

Architecture Development Method Toolbox Pages ................................................................... 26

Architecture Content Model Toolbox Pages ............................................................................. 30

ACM Core .................................................................................................................................. 33

Data Modeling Extension ......................................................................................................... 36

Governance Extension .............................................................................................................. 37

Infrastructure Consolidation Extension .................................................................................... 39

Motivation Extension ................................................................................................................ 40

Process Modeling Extension ..................................................................................................... 41

Services Extension .................................................................................................................... 42

Benefits Toolbox Pages ............................................................................................................ 43

Business Motivation Model Toolbox Pages .............................................................................. 45

Ends Page .................................................................................................................................. 48

Means Page ............................................................................................................................... 49

Impact Page .............................................................................................................................. 51

Assessment Page ...................................................................................................................... 52

Influencers Page ....................................................................................................................... 53

BMM Extended Page ............................................................................................................... 55

Business Logistics Toolbox Pages ............................................................................................ 56

Business Process Toolbox Pages ............................................................................................... 58

Conceptual Framework Toolbox Pages .................................................................................... 59

Enterprise Continuum Toolbox Page ........................................................................................ 61

Organization Structure Toolbox Pages ..................................................................................... 63

Data Map Toolbox Pages ......................................................................................................... 64

Service Model Toolbox Page .................................................................................................... 65

FEAF Business Reference Model Toolbox Page ...................................................................... 67

FEAF Performance Reference Model Toolbox Page ................................................................. 68

FEAF Service Component Reference Model Toolbox Page .................................................. 69

FEAF Technical Reference Model Toolbox Page ...................................................................... 70

Gap Analysis Matrix - TOGAF .................................................................................................. 71

Open the Matrix ....................................................................................................................... 72

Create Gap Elements ............................................................................................................... 74

Gap Analysis Matrix Profiles .................................................................................................... 75
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tagged Values in TOGAF</td>
<td>76</td>
</tr>
<tr>
<td>TOGAF Linked Document Templates</td>
<td>77</td>
</tr>
<tr>
<td>The TOGAF Architecture Development Method</td>
<td>80</td>
</tr>
<tr>
<td>ADM Documentation</td>
<td>81</td>
</tr>
<tr>
<td>The TOGAF Enterprise Continuum</td>
<td>83</td>
</tr>
<tr>
<td>Support For Federal Enterprise Architecture Framework</td>
<td>84</td>
</tr>
<tr>
<td>TOGAF Catalogs</td>
<td>85</td>
</tr>
<tr>
<td>UPDM</td>
<td>86</td>
</tr>
<tr>
<td>Welcome to the MDG Technology for UPDM</td>
<td>87</td>
</tr>
<tr>
<td>MDG Technology for UPDM Copyright Notice</td>
<td>88</td>
</tr>
<tr>
<td>MDG Technology for UPDM Software Product License Agreement</td>
<td>89</td>
</tr>
<tr>
<td>Acknowledgement of Trademarks - UPDM</td>
<td>92</td>
</tr>
<tr>
<td>MDG Technology for UPDM Support</td>
<td>93</td>
</tr>
<tr>
<td>MDG Technology for UPDM System Requirements</td>
<td>94</td>
</tr>
<tr>
<td>Getting Started with MDG Technology for UPDM</td>
<td>95</td>
</tr>
<tr>
<td>Using UPDM</td>
<td>96</td>
</tr>
<tr>
<td>Model Wizard in the MDG Technology for UPDM</td>
<td>97</td>
</tr>
<tr>
<td>UPDM Framework Diagram</td>
<td>98</td>
</tr>
<tr>
<td>UPDM Extensions Menu</td>
<td>100</td>
</tr>
<tr>
<td>UPDM Diagram Types</td>
<td>101</td>
</tr>
<tr>
<td>UPDM Toolboxes</td>
<td>102</td>
</tr>
<tr>
<td>UPDM Stereotypes</td>
<td>104</td>
</tr>
<tr>
<td>Abstract Stereotypes</td>
<td>153</td>
</tr>
<tr>
<td>Quicklinks</td>
<td>159</td>
</tr>
<tr>
<td>Tagged Values for UPDM</td>
<td>160</td>
</tr>
<tr>
<td>Model Validation in MDG Technology for UPDM</td>
<td>161</td>
</tr>
<tr>
<td>Model Validation Rules</td>
<td>162</td>
</tr>
<tr>
<td>Model Views in MDG Technology for UPDM</td>
<td>172</td>
</tr>
<tr>
<td>Glossary</td>
<td>174</td>
</tr>
<tr>
<td>Using Enterprise Architect Elements</td>
<td>175</td>
</tr>
<tr>
<td>The Zachman Framework</td>
<td>177</td>
</tr>
<tr>
<td>Welcome</td>
<td>178</td>
</tr>
<tr>
<td>Zachman Framework Copyright Notice</td>
<td>179</td>
</tr>
<tr>
<td>MDG Technology for Zachman Framework Software Product License Agreement</td>
<td>180</td>
</tr>
<tr>
<td>Acknowledgement of Trademarks - Zachman Framework</td>
<td>182</td>
</tr>
<tr>
<td>Support for the Zachman Framework</td>
<td>183</td>
</tr>
<tr>
<td>Zachman Framework System Requirements</td>
<td>184</td>
</tr>
<tr>
<td>Getting Started with the Zachman Framework</td>
<td>185</td>
</tr>
<tr>
<td>Using the Zachman Framework</td>
<td>186</td>
</tr>
<tr>
<td>The Zachman Framework Interface Diagram</td>
<td>187</td>
</tr>
<tr>
<td>Zachman Framework Model Structure</td>
<td>188</td>
</tr>
<tr>
<td>The Zachman Framework Model Template</td>
<td>190</td>
</tr>
<tr>
<td>Zachman Framework Diagrams</td>
<td>191</td>
</tr>
<tr>
<td>Zachman Framework Diagram Types</td>
<td>192</td>
</tr>
<tr>
<td>The Zachman Framework Toolbox</td>
<td>193</td>
</tr>
<tr>
<td>Business Data Page</td>
<td>195</td>
</tr>
<tr>
<td>Business Process Pages</td>
<td>196</td>
</tr>
<tr>
<td>Business Location Page</td>
<td>197</td>
</tr>
<tr>
<td>Business Motivation Pages</td>
<td>198</td>
</tr>
<tr>
<td>Organization Chart Pages</td>
<td>199</td>
</tr>
<tr>
<td>Business Events Pages</td>
<td>200</td>
</tr>
<tr>
<td>Topic</td>
<td>Page</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Project Viewpoint</td>
<td>265</td>
</tr>
<tr>
<td>Migration Viewpoint</td>
<td>266</td>
</tr>
<tr>
<td>Implementation and Migration Viewpoint</td>
<td>267</td>
</tr>
<tr>
<td>The ArchiMate Language</td>
<td>268</td>
</tr>
<tr>
<td>Language Definitions</td>
<td>269</td>
</tr>
<tr>
<td>Language Structure</td>
<td>276</td>
</tr>
<tr>
<td>Generic Metamodel</td>
<td>277</td>
</tr>
<tr>
<td>Elements</td>
<td>278</td>
</tr>
<tr>
<td>Changing Element Presentation</td>
<td>279</td>
</tr>
<tr>
<td>Motivation Elements</td>
<td>283</td>
</tr>
<tr>
<td>Motivation Example Diagrams</td>
<td>285</td>
</tr>
<tr>
<td>Strategy Layer Elements</td>
<td>288</td>
</tr>
<tr>
<td>Strategy Layer Example Diagrams</td>
<td>289</td>
</tr>
<tr>
<td>Business Layer Elements</td>
<td>290</td>
</tr>
<tr>
<td>Business Layer Example Diagrams</td>
<td>292</td>
</tr>
<tr>
<td>Application Layer Elements</td>
<td>293</td>
</tr>
<tr>
<td>Application Layer Example Diagrams</td>
<td>295</td>
</tr>
<tr>
<td>Technology Layer Elements</td>
<td>296</td>
</tr>
<tr>
<td>Technology Examples</td>
<td>298</td>
</tr>
<tr>
<td>Physical Layer Elements</td>
<td>299</td>
</tr>
<tr>
<td>Physical Examples</td>
<td>300</td>
</tr>
<tr>
<td>Implementation and Migration Layer Elements</td>
<td>301</td>
</tr>
<tr>
<td>Implementation and Migration Example</td>
<td>302</td>
</tr>
<tr>
<td>Composite Elements</td>
<td>303</td>
</tr>
<tr>
<td>Relationships</td>
<td>304</td>
</tr>
<tr>
<td>Relationships Overview</td>
<td>306</td>
</tr>
<tr>
<td>Structural Relationships</td>
<td>309</td>
</tr>
<tr>
<td>Dependency Relationships</td>
<td>310</td>
</tr>
<tr>
<td>Dynamic Relationships</td>
<td>311</td>
</tr>
<tr>
<td>Other Relationships</td>
<td>312</td>
</tr>
<tr>
<td>Relationship Connectors</td>
<td>313</td>
</tr>
<tr>
<td>Alignment between Layers</td>
<td>314</td>
</tr>
<tr>
<td>Customizing the ArchiMate Language</td>
<td>316</td>
</tr>
<tr>
<td>Example Profile</td>
<td>319</td>
</tr>
<tr>
<td>Exchanging ArchiMate Models</td>
<td>320</td>
</tr>
<tr>
<td>Generate Model Exchange File</td>
<td>321</td>
</tr>
<tr>
<td>Import ArchiMate Model Exchange File</td>
<td>323</td>
</tr>
<tr>
<td>Migrate to the Latest ArchiMate Version</td>
<td>325</td>
</tr>
</tbody>
</table>
Architecture Frameworks

Enterprise Architecture is becoming an important discipline, as organizations need to understand the fundamental aspects of their business in order to keep pace with the global market in a continually evolving world. Enterprise Architect has built-in support for all of the important enterprise architecture frameworks and enterprise modeling languages, allowing an enterprise to be modeled from the business goals and drivers down to Cloud-based infrastructure services.

Frameworks and Modeling Languages

<table>
<thead>
<tr>
<th>Framework/Language</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Zachman Framework</td>
<td>The Zachman Framework is a widely used approach for engineering Enterprise Architecture. The Framework is a simple, logical structure that helps in organizing the information infrastructure of the Enterprise.</td>
</tr>
<tr>
<td>TOGAF</td>
<td>The Open Group Architecture Framework (TOGAF) is one of the most widely accepted methods for developing Enterprise Architecture, providing a practical, definitive and proven step-by-step method for developing and maintaining Enterprise Architecture.</td>
</tr>
<tr>
<td>UPDM</td>
<td>The MDG Technology for UPDM tightly integrates with Sparx Systems Enterprise Architect and provides a model-based framework for planning, designing and implementing the Unified Profile for DoDAF and MODAF (UPDM) architectures.</td>
</tr>
<tr>
<td>ArchiMate</td>
<td>ArchiMate® is a common language for describing the construction and operation of business processes, organizational structures, information flows, IT systems and technical infrastructure.</td>
</tr>
<tr>
<td>Extending UML</td>
<td>UML Profiles, MDG Technologies and SDK.</td>
</tr>
</tbody>
</table>
# TOGAF

The Open Group Architecture Framework (TOGAF) is one of the most widely accepted methods for developing enterprise architecture. TOGAF is an open framework, providing a practical, definitive and proven step-by-step method for developing and maintaining enterprise architecture.

## Discussion

The topics described here provide an introduction to, and procedural explanation of, using TOGAF in Enterprise Architect.

<table>
<thead>
<tr>
<th>Section</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>Welcome</td>
<td>This section provides an introduction to TOGAF, and contains the formal documentation defining its use with Enterprise Architect.</td>
</tr>
<tr>
<td>Using TOGAF</td>
<td>Get started with TOGAF, learning about the model structure, templates, diagram types and more.</td>
</tr>
<tr>
<td>TOGAF ADM</td>
<td>The key to TOGAF remains a reliable, practical method - the TOGAF Architecture Development Method (ADM) - for defining business needs and developing an architecture that meets those needs, applying the elements of TOGAF and other architectural assets available to the organization.</td>
</tr>
<tr>
<td>The TOGAF Enterprise Continuum</td>
<td>The TOGAF Enterprise Continuum is a 'virtual repository' of all the architecture assets - models, Patterns, architecture descriptions and other artifacts - that exist both within the enterprise and in the IT industry at large, and that the enterprise considers itself to have available for the development of architectures for the enterprise.</td>
</tr>
<tr>
<td>TOGAF Catalogs</td>
<td>Enterprise Architect helps you to create Model Catalog Artifacts, using the TOGAF-Catalog model Pattern, for:</td>
</tr>
<tr>
<td></td>
<td>• Actors</td>
</tr>
<tr>
<td></td>
<td>• Business Services</td>
</tr>
<tr>
<td></td>
<td>• Organization Units</td>
</tr>
<tr>
<td></td>
<td>• Principles</td>
</tr>
<tr>
<td></td>
<td>Requirements and</td>
</tr>
<tr>
<td>---</td>
<td>-----------------------</td>
</tr>
<tr>
<td></td>
<td>Roles</td>
</tr>
</tbody>
</table>
Welcome to TOGAF in Enterprise Architect

Welcome to The Open Group Architecture Framework (TOGAF) integrated with Enterprise Architect. Using this technology, users of Enterprise Architect benefit from TOGAF within a powerful modeling environment based on open standards.

About TOGAF

The Open Group Architecture Framework is one of the most widely accepted methods for developing enterprise architecture. TOGAF is an open framework, providing a practical, definitive and proven step-by-step method for developing and maintaining enterprise architecture.

The key to TOGAF remains a reliable, practical method - the TOGAF Architecture Development Method (ADM) - for defining business needs and developing an architecture that meets those needs, applying the elements of TOGAF and other architectural assets available to the organization.

TOGAF embodies the concept of the Enterprise Continuum to reflect different levels of abstraction in an architecture development process. In this way TOGAF facilitates understanding and co-operation between actors at different levels. It provides a context for the use of multiple frameworks, models, and architecture assets in conjunction with the TOGAF ADM. By means of the Enterprise Continuum, architects are encouraged to leverage all other relevant architectural resources and assets, in addition to the TOGAF Foundation Architecture, in developing an organization-specific IT architecture.

For detailed information on TOGAF itself, visit the TOGAF website.

Benefits of TOGAF

- Helps align business processes and IT to the business strategies and goals
- Provides support for all the phases in the ADM
- Provides support for OMG's Business Motivation Model
- Provides support for the Architecture Content Model
- Provides support for visual modeling of As-Is and To-Be architecture
- Provides support for modeling all four architecture domains specific to TOGAF (Business, Application, Data and Technology)
- Provides support for the report generation of TOGAF work products
- Provides the Open Group's TOGAF 9 deliverable templates as Linked Document templates
- Provides out-of-box FEAF reference models

TOGAF Features

- A visual clickable Interface for ADM
- Useful starter model to help you become productive quickly
- UML profiles for FEAF Business, Performance, Service and Technical Reference Models
- Efficient relationship management for model artifacts with Enterprise Architect's Relationship Matrix and Hierarchy View
- Links to external files, audit log and report generation features of Enterprise Architect, providing additional capability for maintaining and managing your enterprise architecture
- A TOGAF-specific Glossary for the technology
Getting Started

For instructions on how to start using TOGAF, see *Getting Started with TOGAF* and *Using TOGAF*. 
TOGAF Copyright Notices

TOGAF: Copyright © 2003-2018 X/Open Company Ltd, Trading as The Open Group. All Rights Reserved.

Any organization that intends use the methods, resources, and associated documentation suite known as The Open Group Architecture Framework - TOGAF Version 9 (and all earlier versions) for commercial purposes must apply to The Open Group for a commercial licence. See the Open Group TOGAF web site.
TOGAF Software Product License Agreement

This Software Product License Agreement relates to the separately-purchased MDG Technology for TOGAF for use with the Corporate and Professional Editions of Sparx Systems Enterprise Architect. The MDG Technology for TOGAF integrated with the Ultimate and Unified Editions of Enterprise Architect is covered by the Sparx Systems Enterprise Architect Modelling Tool.

MDG Technology for TOGAF, Enterprise Architect MDG Add-In, Version 3.0.
Copyright © 2008-2020 Sparx Systems Pty Ltd. All Rights Reserved

IMPORTANT-READ CAREFULLY: This End User License Agreement ("EULA") is a legal agreement between YOU as Licensee and SPARX for the SOFTWARE PRODUCT identified above. By installing, copying, or otherwise using the SOFTWARE PRODUCT, YOU agree to be bound by the terms of this EULA. If YOU do not agree to the terms of this EULA, promptly delete the unused SOFTWARE PRODUCT.

The copyright in the SOFTWARE PRODUCT and its documentation is owned by Sparx Systems Pty Ltd, A.B.N 38 085 034 546. Subject to the terms of this EULA, YOU are granted a non-exclusive right for the duration of the EULA to use the SOFTWARE PRODUCT. YOU do not acquire ownership of copyright or other intellectual property rights in any part of the SOFTWARE PRODUCT by virtue of this EULA.

Your use of this software indicates your acceptance of this EULA and warranty.

DEFINITIONS

In this End User License Agreement, unless the contrary intention appears:

- "EULA" means this End User License Agreement
- "SPARX" means Sparx Systems Pty Ltd A.C.N 085 034 546
- "Licensee" means YOU, or the organization (if any) on whose behalf YOU are taking the EULA
- "Registered Edition of MDG Technology for TOGAF" means the edition of the SOFTWARE PRODUCT which is available for purchase from the web site: https://sparxsystems.com/products/mdg/tech/togaf/purchase.html, following a thirty (30) day free evaluation period
- "SOFTWARE PRODUCT" or "SOFTWARE" means MDG Technology for TOGAF, which includes computer software and associated media and printed materials, and may include online or electronic documentation
- "SUPPORT SERVICES" means email-based support provided by SPARX, including advice on usage of the SOFTWARE PRODUCT, investigation of bugs, fixes, repairs of models, if and when appropriate, and general product support
- "SPARX SUPPORT ENGINEERS" means employees of SPARX who provide on-line support services
- "Trial Edition of MDG Technology for TOGAF" means the edition of the SOFTWARE PRODUCT which is available free of charge for evaluation purposes for a period of thirty (30) days

GRANT OF LICENSE

In accordance with the terms of this EULA YOU are granted the following rights:

- To install and use ONE copy of the SOFTWARE PRODUCT or, in its place, any prior version for the same operating system, on a single computer; as the primary user of the computer on which the SOFTWARE PRODUCT is installed, YOU may make a second copy for your exclusive use on either a home or portable computer
- To store or install a copy of the SOFTWARE PRODUCT on a storage device, such as a network server, used only to install or run the SOFTWARE PRODUCT over an internal network
- To make copies of the SOFTWARE PRODUCT for backup, archival and instructional purposes
EVALUATION LICENSE

The Trial Edition of MDG Technology for TOGAF is not free software. Subject to the terms of this agreement, YOU are hereby licensed to use this software for evaluation purposes without charge for a period of thirty (30) days.

Upon expiration of the thirty (30) day evaluation period, the SOFTWARE PRODUCT must be removed from the computer. Unregistered use of MDG Technology for TOGAF after the 30-day evaluation period is in violation of Australian, U.S. and international copyright laws.

SPARX may extend the evaluation period on request and at their discretion.

If YOU choose to use this software after the 30-day evaluation period a license must be purchased (as described at https://sparxsystems.com/products/mdg/tech/togaf/purchase.html). Upon payment of the license fee, YOU will be sent details of where to download the registered edition of MDG Technology for TOGAF and will be provided with a suitable software ‘key’ by email.

ADDITIONAL RIGHTS AND LIMITATIONS

YOU hereby undertake not to sell or sub-license the SOFTWARE PRODUCT other than as expressly authorized by this EULA.

NO WARRANTY. The SOFTWARE PRODUCT is provided "AS IS", without warranty of any kind, and SPARX expressly disclaims all warranties and/or conditions with respect to the SOFTWARE PRODUCT, either express, implied or statutory, including, but not limited to, the implied warranties and/or conditions of merchantability, of satisfactory quality, of fitness for a particular purpose, of accuracy, of quiet enjoyment, and of non-infringement of third party rights.

LIMITATION

Under no circumstances shall SPARX be liable for any incidental, special, indirect or consequential damages arising out of or relating to this license or YOUR use, reproduction, modification, distribution of the SOFTWARE PRODUCT, or any portion thereof, whether under a theory of contract, warranty, strict liability or otherwise, even if the copyright holder has been advised of the possibility of such damages and notwithstanding the failure of essential purpose of any remedy.

TRADEMARKS

All names of products and companies used in this EULA, the SOFTWARE PRODUCT, or the enclosed documentation can be trademarks of their corresponding owners. Their use in this EULA is intended to be in compliance with the respective guidelines and licenses. Windows, Windows XP, Windows 2003 Server, Windows 2008 Server, Windows Vista, Windows 7 and Windows 8 are trademarks of Microsoft®.

GOVERNING LAW

This agreement shall be construed in accordance with the laws of the Commonwealth of AUSTRALIA, in the state of Victoria.
Acknowledgement of Trademarks

Trademarks of Microsoft

- Microsoft®
- Windows®

Trademarks of the OMG

- OMG™
- Object Management Group™
- UML™
- Unified Modeling Language™

Trademarks of The Open Group

- TOGAF™
TOGAF Support

Technical support for modeling through TOGAF in Enterprise Architect is available to registered users of Enterprise Architect in exactly the same way as for Enterprise Architect itself.
TOGAF System Requirements

TOGAF version 9.x runs under these environments:

Operating Systems

- Windows 8
- Windows 7
- Windows Vista
- Windows 2008 Server
- Windows 2003 Server
- Windows XP Service Pack 2

Enterprise Architect Versions

- Enterprise Architect Version 11.1 or later
Getting Started With TOGAF

TOGAF is fully integrated with the Unified and Ultimate Editions of Enterprise Architect, in which it is enabled and ready for use.

If you have the Corporate Edition of Enterprise Architect, you can purchase and install an MDG Technology for TOGAF separately; once you have entered the registration key for the MDG Technology for TOGAF, it is automatically available in and integrated with Enterprise Architect, as for the Unified and Ultimate Editions.

You can use the TOGAF profile in the Professional Edition of Enterprise Architect. However, the Gap Analysis Matrix feature is not available for TOGAF in the Professional Edition.

Access TOGAF

1. Create a new Enterprise Architect project file, and click on the top-level Package.
2. Select the 'Design > Model > Perspectives > Model Wizard' option.
3. In the Model Wizard, select the 'Enterprise Architecture > TOGAF' Perspective and the 'Starter Model' Pattern.
4. Click on the Create Patterns button.

A new base TOGAF model is created in the Browser window, containing the TOGAF Architecture Development Method (ADM) structures and the Enterprise Continuum asset Packages, and displaying the TOGAF-ADM (Interface) diagram.
Using TOGAF

TOGAF provides a model-based framework for planning, designing and implementing the Architecture for an Enterprise. The starter model provided with TOGAF acts as a base upon which you can build the Enterprise Architecture. You can create the appropriate diagrams from the extended Enterprise Architect UML diagram set, using Toolbox pages that support every phase of the TOGAF Interface Diagram. You can also align models across the phases of the Architecture Development Method (ADM) using the Enterprise Architect Relationship Matrix.

Notes

- TOGAF is integrated with the features of Enterprise Architect
- Enterprise Architect provides other Service Oriented Architecture tools such as SOMF and SoaML, and broader architecture modeling tools such as ArchiMate, SPEM and Business Rule Modeling, all of which you can use in conjunction with TOGAF to model and develop your Enterprise Architecture
TOGAF Model Patterns

TOGAF includes a set of model Patterns that you can use to generate separate models within your TOGAF project. These are available through the Model Wizard.

Access

Display the Model Wizard window, using any of the methods outlined here. Once in the Model Wizard window, select the 'Model Patterns' tab, and the 'Enterprise Architecture > TOGAF' Perspective.

In the 'TOGAF Perspective' panel, expand 'TOGAF' and select from the TOGAF Patterns:

- Starter Model (includes both ADM and Enterprise Continuum)
- Architecture Development Method (ADM)
- Enterprise Continuum
- Technical Reference Model
- Catalogs

If you require additional diagrams, then also in the Model Wizard, click on the 'Diagrams' tab and (if necessary) select the 'Enterprise Architecture > TOGAF' Perspective. Then select from the diagram categories:

- FEAF Diagrams (Federal Enterprise Architecture Framework)
- TOGAF_BusinessArchitecture
- TOGAF_DataArchitecture
- TOGAF Diagrams

In the 'Diagram Types' panel, select the required diagram type.

<table>
<thead>
<tr>
<th>Ribbon</th>
<th>Design &gt; Model &gt; Add &gt; Model Wizard</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Design &gt; Model &gt; Perspectives &gt; Model Wizard</td>
</tr>
<tr>
<td>Context Menu</td>
<td>Right-click on Package</td>
</tr>
<tr>
<td>Keyboard Shortcuts</td>
<td>Ctrl+Shift+M</td>
</tr>
<tr>
<td>Other</td>
<td>Browser window Header Bar :</td>
</tr>
</tbody>
</table>
The TOGAF Interface Diagram

In Enterprise Architect, the TOGAF Framework is presented as a predefined model. The model-level diagram of this model structure is the TOGAF Interface diagram, which serves as a user interface for the development of Enterprise Architecture based on TOGAF.

The TOGAF Framework model makes use of UML Packages, which is apparent from the model structure diagram. The Interface diagram itself is a standard UML Package diagram, using custom images.
Double-click on a cell of the Interface diagram to open the model Package and diagram corresponding to that particular ADM phase.
The TOGAF Model Structure

Within the TOGAF Framework model, each ADM phase is modeled as the highest-level Package.
The TOGAF Diagrams

TOGAF provides a number of diagram types to support modeling with TOGAF. These diagrams include:

TOGAF diagrams:
- TOGAF Interface
- Conceptual Framework
- Architecture Content
- Architecture Development Method
- Service Model
- Enterprise Continuum
- Standards Information Base

TOGAF_BusinessArchitecture:
- Benefits
- Business Motivation Model
- Organization Structure
- Business Logistics
- Business Process

TOGAF_DataArchitecture:
- Data Map

FEAF diagrams:
- (FEAF) Business Reference Model
- (FEAF) Service Component Reference Model
- (FEAF) Technical Reference Model
- (FEAF) Performance Reference Model

TOGAF-specific diagrams can be created in the same way as for any other diagram in Enterprise Architect. When you open a TOGAF diagram, Enterprise Architect automatically opens the appropriate Toolbox pages for that diagram.
The TOGAF Toolbox Pages

The MDG Technology For TOGAF Toolbox pages provide elements and relationships for the full range of TOGAF diagrams supported by the Technology.

Access

When you open a TOGAF diagram, Enterprise Architect displays the Toolbox pages that are most useful for that particular diagram type. In addition, the 'Common' and 'Artifacts' pages of UML elements and relationships display, regardless of which diagram is open.

The Diagram Toolbox pages can be docked on either side of the diagram, or free floated on top of the diagram to expose more surface for editing.

<table>
<thead>
<tr>
<th>Ribbon</th>
<th>Design &gt; Diagram &gt; Toolbox: &gt; Specify 'TOGAF' in the 'Find Toolbox Item' dialog</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keyboard Shortcuts</td>
<td>Ctrl+Shift+3 : &gt; Specify 'TOGAF' in the 'Find Toolbox Item' dialog</td>
</tr>
<tr>
<td>Other</td>
<td>You can display or hide the Diagram Toolbox by clicking on the or icons at the left-hand end of the Caption Bar at the top of the Diagram View.</td>
</tr>
</tbody>
</table>
Architecture Development Method Toolbox Pages

Architecture Development Method (ADM) elements are used to define and model the TOGAF specific primitives in all the phases of ADM. You use them to define the scope of the architecture.
# Architecture Development Method Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Architecture Vision</td>
<td>Articulates a vision that enables the business goals, responds to the strategic drivers, conforms with the principles, and addresses the stakeholder concerns and objectives. Tagged Values – ID, Scope, Version</td>
</tr>
<tr>
<td>Architecture Board</td>
<td>Captures the definition for a cross-organization Architecture Board. This is a key element in a successful architecture governance strategy, to oversee the implementation of the strategy. This body should be representative of all the key stakeholders in the architecture, and typically comprises a group of executives responsible for the review and maintenance of the overall architecture. Tagged Values – ID, Authority Limits, Responsibilities</td>
</tr>
<tr>
<td>Baseline Architecture</td>
<td>Captures the very high-level definitions of the Baseline environment from the perspective of business information systems and technology. The scope and level of detail to be defined depends on the extent to which existing architecture elements are likely to be carried over into the Target Architecture. Tagged Values – ID, Type, Version</td>
</tr>
<tr>
<td>Target Architecture</td>
<td>Captures the very high-level definitions of the target environment, from the perspective of business information systems and technology. Tagged Values – ID, Type, Version</td>
</tr>
<tr>
<td>Framework Definition</td>
<td>Provides a textual description of the Framework. Tagged Values – ID, Version</td>
</tr>
<tr>
<td>Request for Architecture Work</td>
<td>Captures the information for the Request for Architecture Work, a major input for the ADM phases. This element is designed as a Document Artifact. On creating a new element of this type, double-click on the element to open the Linked Document and select the 'TOGAF - Request for Architecture Work' template from the list of templates available for the 'Copy Template' option. Tagged Values – ID, Architecturing Organization, Sponsoring Organization</td>
</tr>
<tr>
<td>Statement of Architecture Work</td>
<td>Captures the information for the Statement of Architecture Work, a major output for the ADM phases. This element is designed as a Document Artifact. On creating a new element of this type, double-click on the element to open the Linked Document and select the 'TOGAF – Statement of Architecture Work' template from the list of templates available for the 'Copy Template' option. Tagged Values – ID, Version</td>
</tr>
<tr>
<td>Product</td>
<td>Captures the information on a product produced by the enterprise. Tagged Value – ID</td>
</tr>
<tr>
<td>Work Package</td>
<td>Defines a set of actions that achieve one or more objectives for the business. A work Package can be a part of a project, a complete project, or a program.</td>
</tr>
<tr>
<td>Tagged Values – CapabilityDelivered, WorkPackageCategory, ID, Source, Owner</td>
<td></td>
</tr>
<tr>
<td>---------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><strong>Project</strong>&lt;br&gt;Captures the information to define a planned endeavor undertaken to create a product or service.&lt;br&gt;Tagged Values – ID, FutureDirections, Introduction, ProjectDevelopment, Process Overview, References, Target Architecture(s) Mapping</td>
<td></td>
</tr>
<tr>
<td><strong>Business Driver</strong>&lt;br&gt;Defines the business driver in the 'Name' field.&lt;br&gt;Tagged Values – ID, Version</td>
<td></td>
</tr>
<tr>
<td><strong>Business Scenario</strong>&lt;br&gt;Identifies and clarifies business needs, and thereby derives the business requirements that the architecture development has to address. Creating a business scenario involves these steps:&lt;br&gt;1. Identifying, documenting, and ranking the problem driving the scenario.&lt;br&gt;2. Identifying the business and technical environment of the scenario and documenting it in scenario models.&lt;br&gt;3. Identifying and documenting desired objectives.&lt;br&gt;4. Identifying the human actors (participants) and their place in the business model.&lt;br&gt;5. Identifying computer actors (computing elements) and their place in the technology model.&lt;br&gt;6. Identifying and documenting roles, responsibilities, and measures of success per actor; documenting the required scripts per actor, and the results of handling the situation.&lt;br&gt;7. Checking for 'fitness-for-purpose' and refining only if necessary.&lt;br&gt;A Linked Document template for Business Scenarios is provided by the Technology. To use the template, right-click on the element and select the 'Edit Linked Document' menu option. Select 'TOGAF – Business Scenario/Architecture Vision' for the 'Copy template' option.&lt;br&gt;Tagged Value – ID</td>
<td></td>
</tr>
<tr>
<td><strong>Business Entity</strong>&lt;br&gt;A generic element that captures enterprise resources.&lt;br&gt;Tagged Values – ID, Description</td>
<td></td>
</tr>
<tr>
<td><strong>Goal</strong>&lt;br&gt;Captures what is to be achieved by the enterprise, with specifications defined by the Tagged Values.&lt;br&gt;Tagged Values – Assumption, Critical Success Factor, Goal Type, ID, Key Performance Indicator, Measure, Unit Responsible, Opportunity, Strength, Threat, Weakness</td>
<td></td>
</tr>
<tr>
<td><strong>Objective</strong>&lt;br&gt;Captures the attainable, time-targeted, and measurable target that the enterprise seeks to meet in order to achieve its goals.&lt;br&gt;Tagged Value – ID</td>
<td></td>
</tr>
<tr>
<td><strong>Strategy</strong>&lt;br&gt;Captures the strategy statements for the business plan.&lt;br&gt;Tagged Values – Action Plan, Estimated Budget, Estimated Time Period, ID, Measure, Target Value</td>
<td></td>
</tr>
<tr>
<td><strong>IT Governance Strategy</strong>&lt;br&gt;Defines the strategy statement for IT governance.&lt;br&gt;Tagged Values – ID, Version</td>
<td></td>
</tr>
<tr>
<td>Defines and guides the organization, for the use of all assets and resources across</td>
<td></td>
</tr>
<tr>
<td><strong>Principle</strong></td>
<td>the enterprise. Each Principle should be linked to the relevant business objective and key architecture drivers. Tagged Values – ID, Implications, Rationale, Statement, Type, Version</td>
</tr>
<tr>
<td><strong>Guideline</strong></td>
<td>Captures the Guidelines governing the enterprise and its functions, by providing guidance on the optimal ways to carry out design or implementation activities. Tagged Value – ID</td>
</tr>
<tr>
<td><strong>Asset</strong></td>
<td>Captures the enterprise resources that could be estimated for value. Tagged Values – ID, AssetValue, Description</td>
</tr>
<tr>
<td><strong>Document Asset</strong></td>
<td>A subtype of Asset that captures the important document resources of the enterprise. Tagged Values – ID, AssetValue, Description</td>
</tr>
<tr>
<td><strong>Equipment Asset</strong></td>
<td>A subtype of Asset that captures the equipment resources of the enterprise. Tagged Values – ID, AssetValue, Description</td>
</tr>
<tr>
<td><strong>Architecture</strong></td>
<td>Captures summary views of the Architecture Landscape (that is, the state of the enterprise) at particular points in time. Tagged Values – ID, Category, Source, Owner, Subject Matter, View Point, Level Of Detail, Level Of Abstraction, Accuracy, Version, Maturity</td>
</tr>
<tr>
<td><strong>Solution</strong></td>
<td>Captures the summary views of a solution in place for a specific architecture. Tagged Values – ID, Category, Source, Owner, Subject Matter, Time, Volatility, Version, Maturity</td>
</tr>
<tr>
<td><strong>Architecture Building Block</strong></td>
<td>(ABB) Relates to the Architecture Continuum, and is defined or selected as a result of the application of the ADM. Tagged Values – ID, Description, Owning Organization, Rationale, ServicePortfolio</td>
</tr>
<tr>
<td><strong>Solutions Building Block</strong></td>
<td>(SBB) Relates to the Solutions Continuum, and can be either procured or developed. Tagged Values – ID, Description, Supplier Organization</td>
</tr>
</tbody>
</table>
Architecture Content Model Toolbox Pages

The Architecture Content framework provides a structural model for architectural content that enables the major work products that an architect creates to be consistently defined, structured, and presented.
The elements in each of the Architecture Content Model Toolbox pages are described in separate topics:

- **ACM Core**
- Data Modeling Extension
- Governance Extension
- Infrastructure Consolidation Extension
- Motivation Extension
- Process Modeling Extension
- Services Extension

For information on Architecture Content Model relationships, see the topic *Architecture Content Metamodel Relationships* in the [TOGAF 9 online documentation](https://example.com).
# ACM Core

Elements from the ACM page of the Architecture Content Model Toolbox.

## ACM Core Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
<th>Tagged Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actor</td>
<td>Identifies a person, organization or system with a role that initiates or interacts with activities. Actors can be internal or external to an organization.</td>
<td>ID, Category, Source, Owner, #FTEs, ActorGoal, ActorTasks</td>
</tr>
<tr>
<td>Assumption</td>
<td>Defines a statement of probable fact that has not been fully validated at this stage, due to external constraints.</td>
<td>ID, Rationale, Statement, Type</td>
</tr>
<tr>
<td>Business Constraint</td>
<td>Identifies an external factor that prevents an organization from pursuing particular approaches to meet its goals.</td>
<td>ID</td>
</tr>
<tr>
<td>Business Function</td>
<td>Identifies a factor that delivers business capabilities closely aligned to an organization, but not necessarily explicitly governed by the organization.</td>
<td>ID</td>
</tr>
<tr>
<td>Business Requirement</td>
<td>Defines a quantitative statement of business need that must be met by a particular architecture or work Package.</td>
<td>ID</td>
</tr>
<tr>
<td>Business Service</td>
<td>Identifies a service that supports business capabilities through an explicitly defined interface and is explicitly governed by an organization.</td>
<td>ID, Category, Source, Owner, StandardsClass, StandardCreationDate, LastStandardReviewDate, NextStandardReviewDate, RetireDate</td>
</tr>
<tr>
<td>Capability</td>
<td>Defines a business-focused outcome that is delivered by the completion of one or more work Packages. Using a capability-based planning approach, change activities can be sequenced and grouped in order to provide continuous and incremental business value.</td>
<td>ID, Category, Source, Owner, Increments, BusinessValue</td>
</tr>
<tr>
<td>Data Entity</td>
<td>Defines an encapsulation of data that is recognized by a business domain expert as an entity. Logical data entities can be tied to applications, repositories and services, and can be structured according to implementation considerations.</td>
<td>ID, Category, Source, Owner, PrivacyClassification, RetentionClassification</td>
</tr>
<tr>
<td>Gap</td>
<td>Provides a statement of difference between two states. Used in the context of gap analysis, where the difference between the Baseline and Target Architecture is identified.</td>
<td>ID, Category, Source, Owner</td>
</tr>
</tbody>
</table>
| **Organization Unit** | Defines a self-contained unit of resources with line management responsibility, goals, objectives, and measures. Organizations can include external parties and business partner organizations.  
Tagged Values – ID, PersonIncharge |
| **Principle** | Provides a qualitative statement of intent that should be met by the architecture. This has at least a supporting rationale and a measure of importance.  
Tagged Values – ID, Type, Statement, Rationale, Implications |
| **Process** | Represents the flow of control between or within functions and/or services (depending on the granularity of definition). Processes represent a sequence of activities that together achieve a specified outcome, can be decomposed into sub-processes, and can show operation of a function or service (at the next level of detail). Processes can also be used to link or compose organizations, functions, services, and processes.  
Tagged Values – ID, Category, Source, Owner, StandardsClass, StandardCreationDate, LastStandardReviewDate, NextStandardReviewDate, RetireDate, ProcessCriticality, ProcessVolumetrics, ProcessType |
| **Platform Service** | Defines a technical capability required to provide enabling infrastructure that supports the delivery of applications.  
Tagged Values – ID, Category, Source, Owner, StandardClass |
| **Role** | Defines the usual or expected function of an Actor, or the part somebody or something plays in a particular action or event. An Actor can have a number of roles.  
Tagged Values – ID, Category, Source, Owner, Responsibilities |
| **Work Package** | Identifies a set of actions to achieve one or more objectives for the business. A work Package can be a part of a project, a complete project or a program.  
Tagged Values – ID, Category, Source, Owner, CapabilityDelivered |
| **Application Component** | Provides an encapsulation of application functionality aligned to implementation structure.  
See also: 'Logical Application Component' and 'Physical Technology Component'.  
Tagged Values – ID, Category, Source, Owner, StandardsClass, StandardCreationDate, LastStandardReviewDate, NextStandardReviewDate, RetireDate |
| **Logical Application Component** | Provides an encapsulation of application functionality that is independent of a particular implementation.  
Tagged Values – ID, Category, Source, Owner, StandardsClass, StandardCreationDate, LastStandardReviewDate, NextStandardReviewDate, RetireDate |
| **Technology Component** | Provides an encapsulation of technology infrastructure that represents a class of technology product or specific technology product.  
Tagged Values – ID, Category, Source, Owner, StandardsClass, StandardCreationDate, LastStandardReviewDate, NextStandardReviewDate, RetireDate |
<p>| <strong>Physical Technology</strong> | Defines an instance of a specific technology infrastructure product or technology |</p>
<table>
<thead>
<tr>
<th>Component</th>
<th>infrastructure product</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Tagged Values – ID, Category, Source, Owner, StandardsClass, StandardCreationDate, LastStandardReviewDate, NextStandardReviewDate, RetireDate, ModuleName, ProductName, Vendor, Version</td>
</tr>
</tbody>
</table>
# Data Modeling Extension

Elements from the Data Modeling Extension page of the Architecture Content Model Toolbox.

## Data Modeling Extensions Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logical Data Component</td>
<td>Defines a boundary zone that encapsulates related data entities to form a logical location to be held. Tagged Values – ID, Category, Source, Owner, StandardsClass, StandardCreationDate, LastStandardReviewDate, NextStandardReviewDate, RetireDate</td>
</tr>
<tr>
<td>Physical Data Component</td>
<td>Defines a boundary zone that encapsulates related data entities to form a physical location to be held. Tagged Values – ID, Category, Source, Owner, StandardsClass, StandardCreationDate, LastStandardReviewDate, NextStandardReviewDate, RetireDate</td>
</tr>
</tbody>
</table>
## Governance Extension

Elements from the Governance Extension page of the Architecture Content Model Toolbox.

### Governance Extension Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Measure</strong>&lt;br&gt;Identifies an indicator or factor that can be tracked, usually on an ongoing basis, to determine success or alignment with objectives and goals.&lt;br&gt;Tagged Values – ID, Category, Source, Owner</td>
<td></td>
</tr>
</tbody>
</table>
| Service Quality       | Defines a preset configuration of non-functional attributes that can be assigned to a service or service contract.  
|                      | Tagged Values – ID, Category, Source, Owner |
## Infrastructure Consolidation Extension

Elements from the Infrastructure Consolidation Extension page of the Architecture Content Model Toolbox.

### Infrastructure Consolidation Extension Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
</table>
| Location                          | Represents a place where business activity takes place and can be hierarchically decomposed.  
Tagged Values – ID, Category, Source, Owner |
| Logical Technology Component      | Provides an encapsulation of technology infrastructure that is independent of a particular product. A class of technology product.  
Tagged Values – ID, Category, Source, Owner, StandardsClass, StandardCreationDate, LastStandardReviewDate, NextStandardReviewDate, RetireDate |
| Physical Application Component    | Identifies an application, application module, application service or other deployable component of functionality.  
# Motivation Extension

Elements from the Motivation Extension page of the Architecture Content Model Toolbox.

## Motivation Extension Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business Driver</td>
<td>Defines an external or internal condition that motivates the organization to define its goals.</td>
</tr>
<tr>
<td></td>
<td>Tagged Values – ID, Version</td>
</tr>
<tr>
<td>Goal</td>
<td>Provides a high-level statement of intent or direction for an organization. Typically used to measure success of an organization.</td>
</tr>
<tr>
<td></td>
<td>Tagged Values – ID, Category, Source, Owner</td>
</tr>
<tr>
<td>Objective</td>
<td>Identifies a time-bounded milestone for an organization, to demonstrate progress towards a goal.</td>
</tr>
<tr>
<td></td>
<td>Tagged Values – ID</td>
</tr>
</tbody>
</table>
## Process Modeling Extension

Elements from the Process Modeling Extension page of the Architecture Content Model Toolbox.

### Process Modeling Extension Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Control** | Defines a decision-making step with accompanying decision logic, used to determine the execution approach for a process or to ensure that a process complies with governance criteria.  
Tagged Values – ID, Category, Source, Owner |
| **Event** | Defines an organizational state change that triggers processing events; can originate from inside or outside the organization and can be resolved inside or outside the organization.  
Tagged Values – ID, Category, Source, Owner |
| **Product** | Defines the output generated by the business; that is, the business product of the execution of a process.  
Tagged Values – ID, Category, Source, Owner |
Services Extension

Elements from the Services Extension page of the Architecture Content Model Toolbox.

Services Extension Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IT Service</td>
<td>Defines the automated elements of a business service. An information system service can deliver or support part or all of one or more business services. Tagged Values – ID, Category, Source, Owner, DefinitionText, ContactPoint, Availability, ChargeToUser, DependentSystems, StandardsClass, StandardCreationDate, LastStandardReviewDate, NextStandardReviewDate, RetireDate</td>
</tr>
</tbody>
</table>
Benefits Toolbox Pages

You use the Benefits Toolbox to create elements that represent and depict the opportunities identified in an architecture definition, classified according to their relative size, benefit, and complexity. The resulting Benefits diagram can be used by stakeholders to make decisions on selection, prioritization and sequencing of the identified opportunities.

Benefits Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benefit</td>
<td>An Artifact to model the benefit of an opportunity identified in the architecture definition. Tagged Values – ID, Owner, Source, Category</td>
</tr>
<tr>
<td>Feature</td>
<td>Represents a characteristic of a service or solution. Tagged Values – ID, Owner, Source, Category</td>
</tr>
<tr>
<td>KPI</td>
<td>(Key Performance Indicator) A metric used to define and measure progress towards achieving goals or critical success factors. Tagged Values – ID, Owner, Source, Category</td>
</tr>
<tr>
<td>Objective</td>
<td>A statement of an attainable, time-targeted and measurable target that the enterprise seeks to meet in order to achieve its goals. An Objective quantifies a Goal. Tagged Value – ID</td>
</tr>
<tr>
<td>Outcome</td>
<td>The resulting end state of an event, decision or architecture process. Tagged Values – ID, Owner, Source, Category</td>
</tr>
<tr>
<td>Solution</td>
<td>A statement of an operation or activity that supports the outcome. Tagged Values – ID, Owner, Source, Category</td>
</tr>
<tr>
<td>Potential Impact</td>
<td>See the Help on the 'Impact' Page of the Business Motivation Model toolbox.</td>
</tr>
<tr>
<td>Risk</td>
<td>See the Help on the 'Impact' page of the Business Motivation Model toolbox.</td>
</tr>
<tr>
<td>Reward</td>
<td>See the Help on the 'Impact' Page of the Business Motivation Model toolbox.</td>
</tr>
</tbody>
</table>
Business Motivation Model Toolbox Pages

The Business Motivation Model Toolbox page is based on the OMG specification for the Business Motivation Model (BMM). These elements provide a structure for developing, communicating, and managing business plans in an organized manner.
The elements in each of the Business Motivation Model Toolbox pages are described in separate topics:

- **Ends Page**
• Means Page
• Impact Page
• Assessment Page
• Influencers Page
• BMM Extended Page
## Ends Page

Elements from the 'Ends' page of the Business Motivation Model Toolbox.

### Ends Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>End</td>
<td>Groups 'end' concepts (Vision and Desired Result). An End is something the business seeks to accomplish. It does not include any indication of how it is to be achieved. Tagged Values – ID, Category, Source, Owner</td>
</tr>
<tr>
<td>Vision</td>
<td>Describes the future state of the enterprise, without regard to how it is to be achieved. A Vision is supported or made operative by Missions, and is amplified by Goals. Tagged Value – ID</td>
</tr>
<tr>
<td>Desired Result</td>
<td>Groups 'desired result' concepts (Goal and Objective). A Desired Result is an End that is a state or target that the enterprise intends to maintain or sustain. A Desired Result is supported by Courses of Action. One Desired Result can include other Desired Results and can itself be included in another Desired Result. Tagged Values – ID, Category, Source, Owner</td>
</tr>
<tr>
<td>Goal</td>
<td>A statement about a state or condition of the enterprise to be brought about or sustained through appropriate Means. A Goal amplifies a Vision. Tagged Values – Assumption, Critical Success Factor, Goal Type, ID, Key Performance Indicator, Measure, Unit Responsible, Opportunity, Strength, Threat, Weakness</td>
</tr>
<tr>
<td>Objective</td>
<td>A statement of an attainable, time-targeted and measurable target that the enterprise seeks to meet in order to achieve its goals. An Objective quantifies a Goal. Tagged Value – ID</td>
</tr>
</tbody>
</table>
Means Page

Elements from the 'Means' page of the Business Motivation Model Toolbox.

Means Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Means</td>
<td>Groups 'Means' concepts (Mission, Course of Action and Directive). A Means represents any capabilities that can be exploited to achieve the desired Ends. Tagged Values – ID, Category, Source, Owner</td>
</tr>
<tr>
<td>Mission</td>
<td>Captures the mission statement, policies and values of the enterprise. A Mission indicates the ongoing operational activity of the enterprise, and makes a Vision operative. Tagged Values – ID, Category, Source, Owner</td>
</tr>
<tr>
<td>Course of Action</td>
<td>Groups 'course of action' concepts (Strategy and Tactic). A Course of Action is an approach or plan for configuring some aspect of the enterprise involving things, processes, locations, people, timing or motivation, undertaken to achieve Desired Results. A Course of Action channels efforts towards Desired Results. Courses of Action are governed by Directives. It is also possible for the Courses of Action to be formulated based on Directives. Courses of Action can be realized by Business Processes. One Course of Action can include other Courses of Action, and one Course of Action can be enabled by another Course of Action. Tagged Values – ID, Category</td>
</tr>
<tr>
<td>Strategy</td>
<td>Defines the right approach to achieve a set of Goals, given the environmental constraints and risks. A Strategy usually channels efforts towards those Goals. Tagged Values – Action Plan, Estimated Budget, Estimated Time Period, ID, Measure, Target Value</td>
</tr>
<tr>
<td>Tactic</td>
<td>A Course of Action that represents part of the detailing of a Strategy. A Tactic implements one or more Strategies. Tagged Values – ID, Category</td>
</tr>
<tr>
<td>Directive</td>
<td>Indicates how the Course of Action should, or should not, be carried out. A Directive defines, constrains or liberates some aspect of an enterprise. It is intended to assert business structure or to control or influence the behavior of the business, and is stated in declarative form. Directives govern Courses of Action. A Directive is defined to support the achievement of a Desired Result directly. Tagged Values – ID, Category</td>
</tr>
</tbody>
</table>

Captures the policy definitions followed in the enterprise. A Business Policy is a
| Policy | non-actionable Directive whose purpose is to govern or guide the enterprise. Business Policies provide the basis for Business Rules. Business Policies also govern Business Processes. One Business Policy can include other Business Policies. Tagged Value – ID |
**Impact Page**

Elements from the 'Impact' page of the Business Motivation Model Toolbox.

**Impact Toolbox**

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Potential Impact</td>
<td>Groups the concepts of 'impacts' (Risk and Reward). Each Potential Impact is an evaluation that quantifies or qualifies some aspect of an Assessment in specific terms, types or dimensions. An Assessment identifies some Potential Impacts. A Potential Impact can be significant to an Assessment. Tagged Values – ID, Category, Source, Owner</td>
</tr>
<tr>
<td>Risk</td>
<td>A Potential Impact that indicates the possibility of loss, injury, disadvantage or destruction. Tagged Value – ID</td>
</tr>
<tr>
<td>Reward</td>
<td>A Potential Impact that indicates the probability of gain. Tagged Value – ID</td>
</tr>
</tbody>
</table>
Assessment Page

Elements from the 'Assessment' page of the Business Motivation Model Toolbox.

Assessment Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assessment</td>
<td>A judgment on an Influencer that affects the organization's ability to employ its Means or achieve its Ends. A Directive is motivated by an Assessment. Assessments can also use other Assessments. An Assessment can support the achievement of Ends. Tagged Values – ID, Source, Owner</td>
</tr>
<tr>
<td>Strength</td>
<td>This category of Assessment indicates some advantage or area of excellence within the enterprise that can impact its employment of Means or achievement of Ends. It is modeled as a parameter of the Assessment element. Tagged Value – ID</td>
</tr>
<tr>
<td>Weakness</td>
<td>This category of Assessment indicates some area of inadequacy within the enterprise that can impact its employment of Means or achievement of Ends. It is modeled as a parameter of the Assessment element. Tagged Value – ID</td>
</tr>
<tr>
<td>Opportunity</td>
<td>This category of Assessment indicates that some Influencer can have a favorable impact on the organization's employment of Means or achievement of Ends. It is modeled as a parameter of the Assessment element. Tagged Value – ID</td>
</tr>
<tr>
<td>Threat</td>
<td>This category of Assessment indicates that some Influencer can have an unfavorable impact on the organization's employment of Means or achievement of Ends. It is modeled as a parameter of the Assessment element. Tagged Value – ID</td>
</tr>
</tbody>
</table>
Influencers Page

Elements from the 'Influencers' page of the Business Motivation Model Toolbox.

Influencers Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Influencer</td>
<td>An Influencer element groups the elements influencing an Assessment. The Influencers are those that can impact the enterprise in its employment of Means or achievement of its Ends. This impact has influence that is judged in Assessments. Tagged Values – ID, Category</td>
</tr>
<tr>
<td>External Influencer</td>
<td>An External Influencer element groups the elements having an external influence on an Assessment. External Influencers are those outside an enterprise's organizational boundaries that can impact its employment of Means or achievement of Ends. Tagged Values – ID, Category</td>
</tr>
<tr>
<td>Internal Influencer</td>
<td>An Internal Influencer element groups the elements having an internal influence on an Assessment. Internal Influencers are those from within an enterprise that can impact its employment of Means or achievement of Ends. Tagged Values – ID, Category</td>
</tr>
<tr>
<td>Assumption</td>
<td>An Assumption element captures the assumptions made in information manipulation; assumptions are items of information taken for granted or without proof. Tagged Values – ID, Rationale, Statement, Type</td>
</tr>
<tr>
<td>Competitor</td>
<td>An External Influencer that is an individual or enterprise posing a challenge to the subject enterprise. Tagged Value – ID</td>
</tr>
<tr>
<td>Corporate Value</td>
<td>An ideal, custom or institution that an enterprise promotes or agrees with (either positive or negative). Tagged Value – ID</td>
</tr>
<tr>
<td>Customer</td>
<td>An External Influencer as an individual or enterprise that has investigated, ordered, received or paid for products or services from the subject enterprise. Tagged Value – ID</td>
</tr>
<tr>
<td>Environment</td>
<td>An Environment element is the aggregate of surrounding conditions or Influencers affecting the existence or development of an enterprise. Tagged Value – ID</td>
</tr>
<tr>
<td>Habit</td>
<td>A customary practice or use. Tagged Value – ID</td>
</tr>
<tr>
<td>Infrastructure</td>
<td>An Internal Influencer forming the basic underlying framework or features of a</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------------</td>
<td>-------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>system.</strong></td>
<td>Tagged Value – ID</td>
</tr>
<tr>
<td><strong>Issue</strong></td>
<td>A point in question or a matter that is in dispute as between contending partners.</td>
</tr>
<tr>
<td><strong>Management Prerogative</strong></td>
<td>A right or privilege exercised by virtue of ownership or position in an enterprise.</td>
</tr>
<tr>
<td><strong>Partner</strong></td>
<td>An External Influencer as an enterprise that shares risks and profit with the subject enterprise (or is associated with the subject enterprise to share risks and profit) because this is mutually beneficial.</td>
</tr>
<tr>
<td><strong>Principle</strong></td>
<td>Defines and guides the organization, for use of all assets and resources across the enterprise. Each Principle should be linked to the relevant business objective and key architecture drivers.</td>
</tr>
<tr>
<td><strong>Regulation</strong></td>
<td>An External Influencer as an order prescribed by an authority such as a government body or the management of an enterprise.</td>
</tr>
<tr>
<td><strong>Resource</strong></td>
<td>An internal Influencer as a resource available for carrying out the business of an enterprise, applying its influence especially by way of its quality.</td>
</tr>
<tr>
<td><strong>Technology</strong></td>
<td>An External Influencer as the role of technology, including its developments and limitations — there could be prerequisites for use of technology, or an enterprise activity that technology enables or restricts.</td>
</tr>
<tr>
<td><strong>Stakeholder</strong></td>
<td>Captures the actors interested and involved in the enterprise.</td>
</tr>
<tr>
<td><strong>Standard</strong></td>
<td>Defines the standards followed in the enterprise.</td>
</tr>
<tr>
<td><strong>Supplier</strong></td>
<td>An External Influencer as an individual or enterprise that can furnish or provide products or services to the subject enterprise.</td>
</tr>
</tbody>
</table>
BMM Extended Page

Elements from the 'BMM Extended' page of the Business Motivation Model Toolbox.

### BMM Extended Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organization Unit</td>
<td>Represents any recognized association of people in the context of the enterprise. In a hierarchical structure, it might be the corporation, a division, a group or a team. Tagged Values – ID, PersonInCharge</td>
</tr>
<tr>
<td>Liability</td>
<td>A Liability is a reservation of actual resources (materials, finished goods, people's time, cash) to meet commitments. A Liability can be discharged by Courses of Action, can be the responsibility of Organization Units, and can claim Resources. Tagged Value – ID</td>
</tr>
<tr>
<td>Asset</td>
<td>An Asset is something of value owned by the enterprise. Tagged Values – ID, Description, AssetValue</td>
</tr>
<tr>
<td>Offering</td>
<td>An Offering is a Fixed Asset that is a specification of a product or service that can be supplied by the enterprise. An Offering can be defined by Courses of Action, can be delivered by Business Processes, can require Resources and can use Fixed Assets. Tagged Value – ID</td>
</tr>
<tr>
<td>Business Process</td>
<td>A function or behavior of the Enterprise or part of the Enterprise. A Business Process is the responsibility of an Organization Unit, realizes Courses of Action, is guided by Business Rules, is governed by Business Policies, can deliver Offerings and can manage Assets. Tagged Values – ID, Description, ProcessType</td>
</tr>
<tr>
<td>Fixed Asset</td>
<td>A Fixed Asset is an Asset that is maintained over time and reused. A Fixed Asset can be used by Offerings and can provide Resources. Tagged Values – ID, AssetValue</td>
</tr>
</tbody>
</table>
Business Logistics Toolbox Pages

**Item** | **Description**
--- | ---
Branch Office | Models a Business Location as a Branch Office.
Client Place | Models a Business Location as a Client Place.
Head Quarters | Models a Business Location as a Head Quarters.
Business Location | Models the location from which the business operates.
Office Block | Models a Business Location as an Office Block.
Sales Agent | Models a Business Location as a Sales Agent.
Supplier | Models a Business Location as a Supplier.
Communicate | Indicates that a business location communicates directly with another business location.
Internet | Indicates that the means of communication is the World Wide Web.
Snail Mail | Indicates that the means of communication is the postal system or courier services.
Phone | Indicates that the means of communication is the telephone.
<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>In Person</td>
<td>Indicates that the means of communication is direct person-to-person.</td>
</tr>
<tr>
<td>Intranet</td>
<td>Indicates that the means of communication is the local intranet or WAN.</td>
</tr>
</tbody>
</table>

**Notes**

- Elements and connectors common to Enterprise Architect UML and Extended diagrams are documented in the [Object Toolbox](#) section.
Business Process Toolbox Pages

![Business Process Toolbox](image)

Business Process Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actor</td>
<td>Models a stakeholder or any other human resource of the Enterprise.</td>
</tr>
<tr>
<td>Decision</td>
<td>Indicates point of conditional progression where a business decision is taken.</td>
</tr>
<tr>
<td>Business Function</td>
<td>A major function performed by the Enterprise or a part of the Enterprise.</td>
</tr>
<tr>
<td>Business Process</td>
<td>A function or behavior of the Enterprise or part of the Enterprise.</td>
</tr>
<tr>
<td>Business Entity</td>
<td>A generic element to capture Enterprise resources.</td>
</tr>
<tr>
<td>Invokes</td>
<td>A relationship that defines the invocation of a business process.</td>
</tr>
</tbody>
</table>

Notes

- Elements and connectors common to Enterprise Architect UML and Extended diagrams are documented in the [Object Toolbox](#) section
Conceptual Framework Toolbox Pages

The Conceptual Framework Elements are used to model the architectural descriptions and to establish concepts for architectural thinking. The Toolbox element design is based on IEEE standard 1471 - 2000.

### Conceptual Framework Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mission</td>
<td>Captures the mission statement, policies and values of the enterprise. Tagged Value – ID</td>
</tr>
<tr>
<td>Environment</td>
<td>Defines the developmental, operational and programmatic context of the system for the purpose of Enterprise Engineering work. Tagged Value – ID</td>
</tr>
<tr>
<td>System</td>
<td>Captures details of a working component of the enterprise. System includes, for example, application, system, platform, system -of-systems, enterprise and product line. Tagged Value – ID</td>
</tr>
<tr>
<td>Architecture</td>
<td>Captures the definition of the Architecture work. Tagged Value – ID</td>
</tr>
<tr>
<td>Stakeholder</td>
<td>Captures the actors interested and involved in the enterprise. Tagged Value – ID</td>
</tr>
<tr>
<td>Architectural Description</td>
<td>Captures the architectural descriptions and identifies the system’s stakeholders and their concerns. Tagged Value – ID</td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Rationale</td>
<td>Captures the statement of purpose for the Architectural Description.</td>
</tr>
<tr>
<td>Concern</td>
<td>Forms the basis for completeness. An Architectural Description addresses all stakeholders’ concerns, and each Concern is addressed by an Architectural View</td>
</tr>
<tr>
<td>Viewpoint</td>
<td>A Pattern for constructing Views – Viewpoints define the rules on Views. Each View corresponds to exactly one Viewpoint. Tagged Value – ID</td>
</tr>
<tr>
<td>View</td>
<td>A representation of a whole system from the perspective of a set of Concerns. A View can contain one or more architectural models, so the View can use multiple notations.</td>
</tr>
<tr>
<td>Library Viewpoint</td>
<td>Captures a collection of categorized Viewpoints. Tagged Value – ID</td>
</tr>
<tr>
<td>Model</td>
<td>Defines and represents a model. Tagged Value – ID</td>
</tr>
</tbody>
</table>

**Notes**

- Elements and connectors common to Enterprise Architect UML and Extended diagrams are documented in the [Object Toolbox](#) section.
Enterprise Continuum Toolbox Page

Enterprise Continuum elements are used to model the Architecture Continuum and Solutions Continuum of an enterprise. Using these elements you can create Architecture Building Blocks or Solutions Building Blocks by mapping to the appropriate architecture models or solution models (as diagrams, elements and models).

Enterprise Continuum Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enterprise Continuum</td>
<td>A Package that models the Enterprise Continuum.</td>
</tr>
<tr>
<td></td>
<td>Tagged Values – ID, Architecturing Organization, Sponsoring Organization</td>
</tr>
<tr>
<td>Architecture Continuum</td>
<td>A Package that models the Architecture Continuum.</td>
</tr>
<tr>
<td>Solutions Continuum</td>
<td>A Package that models the Solutions Continuum.</td>
</tr>
<tr>
<td>Architecture</td>
<td>Captures summary views of the Architecture Landscape (such as the state of the enterprise) at particular points in time.</td>
</tr>
<tr>
<td></td>
<td>Tagged Values – ID, Category, Source, Owner, Subject Matter, View Point, Level Of Detail, Level Of Abstraction, Accuracy, Version, Maturity</td>
</tr>
<tr>
<td>Solution</td>
<td>Captures the summary views of a solution in place for a specific architecture.</td>
</tr>
<tr>
<td></td>
<td>Tagged Values – ID, Category, Source, Owner, Subject Matter, Time, Volatility, Version, Maturity</td>
</tr>
<tr>
<td>Architecture Building Block</td>
<td>Relates to the Architecture Continuum, and is defined or selected as a result of the application of the ADM.</td>
</tr>
<tr>
<td></td>
<td>Tagged Values – ID, Description, Owning Organization, Rationale, ServicePortfolio</td>
</tr>
<tr>
<td>Solutions Building Block</td>
<td>Relates to the Solutions Continuum, and can be either procured or developed.</td>
</tr>
<tr>
<td></td>
<td>Tagged Values – ID, Description, Supplier Organization</td>
</tr>
<tr>
<td>Connector</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ABB Mapping</td>
<td>Connector to map the architectural models and artifacts to the Architecture Building Blocks.</td>
</tr>
<tr>
<td>SBB Mapping</td>
<td>Connector to map the solution models and artifacts to the Solutions Building Blocks.</td>
</tr>
<tr>
<td>Guides</td>
<td>Connector to represent guides relationships. Architecture Building Blocks guide the development of Solutions Building Blocks.</td>
</tr>
<tr>
<td>Supports</td>
<td>Connector to represent supports relationships. Solutions Building Blocks support the development of other Solutions Building Blocks.</td>
</tr>
<tr>
<td>Leverage Direction</td>
<td>Connector to represent the direction of leveraging of architecture and solution components.</td>
</tr>
</tbody>
</table>
Organization Structure Toolbox Pages

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Board of Directors</td>
<td>Captures the details of the board of directors.</td>
</tr>
<tr>
<td>StakeHolder</td>
<td>Captures stakeholders of the enterprise.</td>
</tr>
<tr>
<td>External Organization</td>
<td>Captures any external business unit that is not under direct control of the enterprise, but has a relationship with the enterprise.</td>
</tr>
<tr>
<td>Organization Unit</td>
<td>Captures any business unit that is under direct control of the enterprise.</td>
</tr>
<tr>
<td>Personnel</td>
<td>Captures the details of personnel in an enterprise.</td>
</tr>
<tr>
<td>In Contract</td>
<td>Captures the contract-based relationships between business units.</td>
</tr>
<tr>
<td>Works For</td>
<td>Captures the details of team links; for example, Stakeholder 1 works for Organization Unit 1.</td>
</tr>
<tr>
<td>Supervise</td>
<td>Captures process supervision details.</td>
</tr>
<tr>
<td>Control</td>
<td>Captures Unit in charge or Person in charge information.</td>
</tr>
</tbody>
</table>

Notes

- Elements and connectors common to Enterprise Architect UML and Extended diagrams are documented in the Object Toolbox section
Data Map Toolbox Pages

Data Map Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Principal Entity</td>
<td>A business entity that forms a resource of the enterprise.</td>
</tr>
<tr>
<td>Intersecting Entity</td>
<td>Normalizes the many-to-many relationship between principal entities.</td>
</tr>
<tr>
<td>Structure Entity</td>
<td>Captures potential knowledge base entities.</td>
</tr>
</tbody>
</table>

Notes

- Elements and connectors common to Enterprise Architect UML and Extended diagrams are documented in the [Object Toolbox](#) section
Service Model Toolbox Page

Service Model elements are used to build a conceptual framework that describes the IT Service infrastructure of the enterprise.

### Service Model Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IT Service</td>
<td>Captures the IT capability offered as a consumable entity that is managed by the enterprise. Tagged Values – ID, DefinitionText, Owner, Availability, Charge_to_User, ContactPoint, Dependent_Systems</td>
</tr>
<tr>
<td>IT Service Domain</td>
<td>Categorizes IT services. Tagged Values – ID, Description</td>
</tr>
<tr>
<td>IT Service Component</td>
<td>Captures a set of capabilities that might be exposed through the technology interface. Tagged Values – ID, Rationale</td>
</tr>
<tr>
<td>IT Service Portfolio</td>
<td>A Document Artifact that captures the information required to describe an IT service portfolio. Tagged Values – ID</td>
</tr>
<tr>
<td>Capability</td>
<td>A business-focused outcome that is delivered by the completion of one or more work Packages. Using a capability-based planning approach, change activities can be sequenced and grouped in order to provide continuous and incremental business value. Tagged Values – ID, Category, Increments, Business Value, Source, Owner</td>
</tr>
<tr>
<td>System</td>
<td>Captures details of a working component of the enterprise. System includes things such as application, system, platform, system-of-systems, enterprise and product line. Tagged Values – ID</td>
</tr>
<tr>
<td>Asset</td>
<td>Captures the enterprise resources that could be estimated for value. Tagged Values – ID, AssetValue, Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Document Asset</td>
<td>Subtype of Asset that captures the important document resources of the enterprise. Tagged Values – ID, AssetValue, Description</td>
</tr>
<tr>
<td>Equipment Asset</td>
<td>Subtype of Asset that captures the equipment resources of the enterprise. Tagged Values – ID, AssetValue, Description</td>
</tr>
</tbody>
</table>
FEAF Business Reference Model Toolbox Page

The FEAF Business Reference Model (BRM) provides a framework facilitating a functional (rather than organizational) view of the enterprise’s lines of business (LoBs), including its internal operations and its services.

### FEAF Business Reference Model Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BRM</td>
<td>A Package in which to capture the Business Reference Model (BRM). Tagged Values – Version</td>
</tr>
<tr>
<td>Business Area</td>
<td>The high-level organizing layer of the BRM, capturing high-level categories relating to the business purpose and objectives. Tagged Values – BusinessAreaID, Definition</td>
</tr>
<tr>
<td>Business Line</td>
<td>Captures the lines of business of the enterprise. Tagged Values – BusinessLineID, Definition, Referencing Business Area</td>
</tr>
<tr>
<td>SubFunction</td>
<td>Represents the lowest level of granularity in the BRM, grouping functionalities related to each line of business. Tagged Values – SubFunctionID, Definition, Referencing BusinessLine, Referencing Business Area</td>
</tr>
</tbody>
</table>
FEAF Performance Reference Model Toolbox Page

The FEAF Performance Reference Model (PRM) Toolbox page is designed to conform to the specification of the FEAF-PRM framework. The PRM is a framework for performance measurement providing common output measurements throughout the enterprise. It enables agencies to better manage the business at a strategic level, by providing a means for using an agency’s Enterprise Architect to measure the success of IT investments and their impact on strategic outcomes.

The FEAF Performance Reference Model (PRM) facilitates resource-allocation decisions based on comparative determinations of which programs and organizations are more efficient and effective.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRM</td>
<td>A Package to capture the Performance Reference Model.</td>
</tr>
<tr>
<td></td>
<td>Tagged Values – Version</td>
</tr>
<tr>
<td>Measurement Area</td>
<td>The high-level organizing layer of the PRM, capturing aspects of performance at the output levels. This layer is directly linked to the performance objectives established at the agency and program levels.</td>
</tr>
<tr>
<td></td>
<td>Tagged Values – MeasurementAreaID, Definition</td>
</tr>
<tr>
<td>Measurement Category</td>
<td>Categorizes the measurement area with respect to the attribute or characteristic to be measured.</td>
</tr>
<tr>
<td></td>
<td>Tagged Values – MeasurementCategoryID, Definition, Referencing Measurement Area</td>
</tr>
<tr>
<td>Measurement Grouping</td>
<td>Further refines Measurement Categories into specific types of Measurement Indicator.</td>
</tr>
<tr>
<td></td>
<td>Tagged Values – MeasurementGroupingID, Definition, Referencing Measurement Category</td>
</tr>
<tr>
<td>Measurement Indicator</td>
<td>Captures the specific measures.</td>
</tr>
<tr>
<td></td>
<td>Tagged Values – MeasurementIndicatorID, Definition, Referencing Measurement Grouping</td>
</tr>
</tbody>
</table>
The FEAF Service Component Reference Model (SRM) is a business-driven, functional framework classifying Service Components according to how they support business and performance objectives. The model aids in recommending service capabilities to support the reuse of business components and services across the enterprise. The SRM should be structured across horizontal service areas that, independent of the business functions, can provide a leverage-able foundation for reuse of applications, application capabilities, components, and business services.

### FEAF Service Component Reference Model Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRM</td>
<td>A Package to capture the Service Component Reference Model. Tagged Values – Version</td>
</tr>
<tr>
<td>Service Domain</td>
<td>Captures a high-level view of the services and capabilities that support enterprise and organizational processes and applications. Tagged Values – ServiceDomainID, Definition</td>
</tr>
<tr>
<td>Service Type</td>
<td>Groups similar capabilities in support of the domain, providing an additional layer of categorization that defines the context of a specific capability component within a given domain. Tagged Values – ServiceTypeID, Definition, Referencing Service Domain</td>
</tr>
<tr>
<td>Service Component</td>
<td>Captures a set of capabilities that might be exposed through a business or technology interface. Service Components are 'building blocks' to deliver the information management capability to the business. Tagged Values – ServiceComponentID, Definition, Referencing Service Domain, Referencing Service Type</td>
</tr>
</tbody>
</table>
FEAF Technical Reference Model Toolbox Page

The FEAF Technology Reference Model (TRM) is a component-driven, technical framework categorizing the standards and technologies to support and enable the delivery of Service Components and capabilities.

### FEAF Technical Reference Model Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRM</td>
<td>A Package to capture the Technology Reference Model. Tagged Value – Version</td>
</tr>
<tr>
<td>Service Area</td>
<td>Represents a technical tier supporting the secure construction, exchange, and delivery of a Service Component. Tagged Values – ServiceAreaID, Definition</td>
</tr>
<tr>
<td>Service Category</td>
<td>Classifies a lower level of technology and standard with respect to the business or technology function it serves. Tagged Values – ServiceCategoryID, Definition, Referencing Service Area</td>
</tr>
<tr>
<td>Service Standard</td>
<td>Defines a standard and technology that supports a Service Category. Tagged Values – ServiceStandardID, Definition, Referencing Service Category</td>
</tr>
</tbody>
</table>
Gap Analysis Matrix - TOGAF

The Specification document for TOGAF states:

'Gap analysis is widely used in the TOGAF Architecture Development Method (ADM) to validate an architecture that is being developed. The basic premise is to highlight a shortfall between the Baseline Architecture and the Target Architecture; that is, items that have been deliberately omitted, accidentally left out, or not yet defined.'

TOGAF provides a Gap Analysis Matrix that you can use to:

- Identify gaps between the baseline and target
- Create Gap elements (if any gaps are identified) in the repository, which can later be addressed and assigned as tasks; the Gap elements can then be used to prioritize activities
- Create and manage Gap Analysis Matrix profiles

Notes

- This feature is not available in the Professional Edition of Enterprise Architect
Open the Matrix

Access

| Ribbon | Design > Impact > Gap Analysis |

Example

This Gap Analysis Matrix example is from the TOGAF Specification; it illustrates Gap Analysis for Architecture Building Blocks (ABBs) that are services from the Network Services category.

![Gap Analysis Matrix example](image-url)
Using the Gap Analysis Matrix

The 'Filter' fields list all the stereotypes that can be shown in the matrix; use these fields to set a filter for each of the target and baseline Architectures.

After setting the filters, click on the button to the right of the 'Target Architecture' and 'Baseline Architecture' fields, and browse for and select the target Architecture Package and baseline Architecture Package.

Click on the Refresh button; the matrix lists the elements having the stereotypes you set in the 'Filter' fields. The target Architecture elements are listed horizontally as column headings, and the baseline Architecture elements are listed vertically as row titles. If you double-click on the row or column headers containing the baseline or target elements, the corresponding 'Properties' dialog displays.

To locate an object in the Browser window, right-click on it and select the 'Find in Project Browser' option.

In the cells at the intersection of the target element columns and baseline element rows, you can create and edit Gap Analysis Notes. To edit the notes double-click on the cell, or right-click and select the 'Edit Notes' option.

Any elements that are not in the target Architecture but are available in the baseline Architecture must be addressed as Gap elements in the last column, called 'Missing / Eliminated'. Any elements that are in the target Architecture but not in the baseline Architecture must be addressed as Gap elements in the last row, called 'New'.

In the example:

- Broadcast Services and Shared Screen Services are present in the baseline Architecture but missing from the target Architecture; therefore, you must create appropriate Gap elements in the 'Missing / Eliminated' column, the last column of the matrix
- Mailing List Services is not in the baseline Architecture but it is in the target Architecture, meaning that the service is a new one in the target Architecture and it must be procured or developed; you must create a corresponding Gap element in the 'New' row, the last row of the matrix

Notes

- Locating the baseline/target element in the 'Project' tab of the Browser window with the Traceability Window open, and then switching to the 'Details' tab of the Inspector window, helps improve gap analysis, as it shows all the elements and details such as Tagged Values that are linked to the element; for example, if an Architecture Building Block (ABB) is missing in the target Architecture, you can see what other processes and tasks depend on this ABB and what processes are impacted, which can also help you to decide whether or not an ABB must be enhanced in the target Architecture
Create Gap Elements

Create a Gap element to model an identified gap

1. Right-click on the cell and select the 'Create Gap Element' option. The 'Browse Project' dialog displays.
2. Select the Package in which to create the Gap element and click on the OK button. A Gap element is created in the selected Package and its 'Properties' dialog displays; enter the element name and other required properties.
3. Select the 'Tagged Values' tab of the 'Properties' dialog and set the Tagged Values listed under 'Gap Element Tagged Values'.
4. Click on the OK button. The name and category of the Gap element displays in the selected matrix cell.

Gap Element Tagged Values

If you intend to use a Gap element that is already available in the model, right-click on the appropriate cell in the 'Missing / Eliminated' column or 'New' row and select the 'Link to Existing Gap Element' option. The 'Select Classifier' dialog displays, from which you select the existing Gap element.

Once you have created a Gap element, you can right-click on its cell and select from these context menu options:

- 'Edit Gap Element' to open the 'Properties' dialog of the Gap element and edit its properties
- 'Locate in Project Browser' to find and highlight the Gap element in the Browser window
- 'Remove Gap Element Link' to delete the link to the element in that cell (the element still exists in its parent Package)
- 'Delete Gap Element' to delete the element from the model; this action cannot be undone

<table>
<thead>
<tr>
<th>Tagged Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>The unique identifier for the architecture object.</td>
</tr>
<tr>
<td>Owner</td>
<td>The owner of the architecture object.</td>
</tr>
<tr>
<td>Source</td>
<td>The location/source from which the information is collected.</td>
</tr>
<tr>
<td>Category</td>
<td>The categorization of the Gap. This can have any one of the values:</td>
</tr>
<tr>
<td></td>
<td>• Intentionally eliminated</td>
</tr>
<tr>
<td></td>
<td>• Unintentionally eliminated</td>
</tr>
<tr>
<td></td>
<td>• New – To be produced or developed</td>
</tr>
<tr>
<td></td>
<td>• To be enhanced</td>
</tr>
<tr>
<td>RefBaseline Architecture</td>
<td>The name of the baseline Architecture artifact that is linked to the Gap element. If the gap is to point to a missing element, this tag has the value of the baseline artifact that is missing.</td>
</tr>
<tr>
<td>RefTarget Architecture</td>
<td>The name of the target Architecture artifact that is linked to the Gap element. If the gap points to a new artifact that is required for the target Architecture, this tag has the value of the new target artifact.</td>
</tr>
</tbody>
</table>
Gap Analysis Matrix Profiles

On the Gap Analysis Matrix, you can create and manage profiles to save commonly-used combinations of target Architectures and stereotypes.

To work on Gap Analysis Matrix profiles, click on the Options button in the top right corner of the matrix. A submenu displays, listing options to:

- Create a profile of the current matrix settings
- Update the currently-selected profile in the 'Profile' field
- Delete the currently-selected profile

The 'Profile' field drop-down list shows all the saved Gap Analysis Matrix profiles.
Tagged Values in TOGAF

TOGAF makes extensive use of Tagged Values for assigning custom properties to the various elements specific to TOGAF. When creating or viewing a TOGAF model, it is recommended that you keep the Properties window docked and visible at all times, with the TOGAF section expanded.

Synchronize Tagged Values

From time to time you might need to add missing Tagged Values to all elements in the model that require them, such as:

- Whenever you create a new element by any means other than directly dropping the element from the TOGAF Toolbox pages
- Before using a new version of the Technology, to update the Tagged Values of elements in existing models to the latest version of the TOGAF profile

You can do this using the 'Synchronize Stereotype' option on the icons in the TOGAF pages of the Diagram Toolbox.
TOGAF Linked Document Templates

Enterprise Architect contains a set of Linked Document templates that are specific to TOGAF.
You can also select these templates from the drop-down list in the 'New Linked Document from Template' dialog; scroll down to the 'Technology Templates' list.

The Linked Document templates are provided by The Open Group, contingent on this text being displayed in any...
documentation of the templates:
"The Open Group TOGAF 9 templates and examples.
Copyright (c) 2010 The Open Group.
The Open Group gratefully acknowledges Capgemini for contributing these templates and examples.
Permission to use, copy, modify, and distribute this set of examples and templates (the 'distribution') for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of The Open Group not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. The Open Group makes no representations about the suitability of this distribution for any purpose. It is provided "as is" without express or implied warranty.
THE OPEN GROUP DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS DISTRIBUTION INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL THE OPEN GROUP BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS DISTRIBUTION.
TOGAF is a trademark of The Open Group."
The TOGAF Architecture Development Method

The key to TOGAF remains a reliable, practical method - the TOGAF Architecture Development Method (ADM) - for defining business needs and developing an architecture that meets those needs, applying the elements of TOGAF and other architectural assets available to the organization.

TOGAF embodies the concept of the Enterprise Continuum to reflect different levels of abstraction in an architecture development process. In this way TOGAF facilitates understanding and co-operation between actors at different levels. It provides a context for the use of multiple frameworks, models, and architecture assets in conjunction with the TOGAF ADM. By means of the Enterprise Continuum, architects are encouraged to leverage all other relevant architectural resources and assets, in addition to the TOGAF Foundation Architecture, in developing an organization-specific IT architecture.

Key Points About the ADM

The ADM is iterative over the whole process, between phases and within phases; for each iteration of the ADM, a fresh decision must be taken on:

- The breadth of coverage of the enterprise to be defined
- The level of detail to be defined
- The extent of the time horizon aimed at, including the number and extent of any intermediate time horizons
- The architectural assets to be leveraged in the organization's Enterprise Continuum, including:
  - Assets created in previous iterations of the ADM cycle within the enterprise
  - Assets available elsewhere in the industry (such as other frameworks, systems models and vertical industry models)

These decisions must be made on the basis of a practical assessment of resource and competence availability, and the value that can realistically be expected to accrue to the enterprise from the chosen scope of the architecture work.

As a generic method, the ADM is intended to be used by enterprises in a wide range of different geographies and applied in different vertical sectors/industry types. As such it can be - but does not necessarily have to be - tailored to specific needs. For example, it can be used:

- In conjunction with the set of deliverables of another framework, where these are more appropriate for a specific organization; many US federal agencies have developed individual frameworks that define the deliverables specific to their particular departmental needs
- In conjunction with the well-known Zachman Framework, which is an excellent classification scheme but which lacks an openly available, well-defined methodology
ADM Documentation

The Architecture Development Method (ADM) has ten Phases, as identified here. The approach and complete description of each Phase are provided in the TOGAF 9 documentation available on The Open Group website, to identify the objectives, inputs, steps and outputs of each phase.

Preliminary Phase: Framework and Principles

The Preliminary Phase is about defining 'where, what, why, who, and how we do architecture' in the enterprise concerned. The main aspects are:

- Defining the enterprise
- Identifying key drivers and elements in the organizational context
- Defining the requirements for architecture work
- Defining the architecture principles that will inform any architecture work
- Defining the framework to be used
- Defining the relationships between management frameworks
- Evaluating the enterprise architecture maturity

Phase A: Architecture Vision

Architecture Vision starts with receipt of a Request for Architecture Work from the sponsoring organization to the architecture organization. During this phase, you define the architecture scope, how to create the vision, and obtain approvals.

Phase B: Business Architecture

Business Architecture is the first architecture activity that must be undertaken, if not catered for already in other organizational processes (such as enterprise planning, strategic business planning or business process re-engineering).

Phase C: Information Systems Architectures

In this phase you develop the Information Systems Architectures, including the Data and Applications Architectures. Detailed steps for Phase C are given separately for each architecture domain:

- Data Architecture
- Applications Architecture

Phase D: Technology Architecture

The steps within the Technology Architecture phase are:

- Select reference models, viewpoints, and tools
- Develop Baseline Technology Architecture Description
- Develop Target Technology Architecture Description
• Perform gap analysis
• Define roadmap components
• Resolve impacts across the Architecture Landscape
• Conduct formal stakeholder review
• Finalize the Technology Architecture
• Create Architecture Definition Document

Phase E: Opportunities and Solutions

In the Opportunities and Solutions phase you identify the parameters of change, the major phases along the way, and the top-level projects to be undertaken in moving from the current environment to the target.

Phase F: Migration Planning

During the Migration Planning phase you sort the various implementation projects into priority order. Activities include assessing the dependencies, costs and benefits of the various migration projects.

Phase G: Implementation Governance

During the Implementation Governance phase you bring together all the information for successful management of the various implementation projects.

Phase H: Architecture Change Management

In the Architecture Change Management phase you establish an architecture change management process for the new enterprise architecture baseline.

ADM Architecture Requirements Management

The ADM is continuously driven by the Architecture Requirements Management process.
The TOGAF Enterprise Continuum

It is simplest to think of the Enterprise Continuum as a 'virtual repository' of all the architecture assets - models, Patterns, architecture descriptions and other artifacts - that exist both within the enterprise and in the IT industry at large, and that the enterprise considers itself to have available for the development of architectures for the enterprise.

Examples of ‘assets within the enterprise’ are the deliverables of previous architecture work that are available for re-use.

Examples of ‘assets in the IT industry at large’ are the wide variety of industry reference models and architecture Patterns that exist and that are continually emerging, including those that are:

- Highly generic, such as TOGAF's own Technical Reference Model (TRM)
- Specific to certain aspects of IT, such as a web services architecture, or a generic manageability architecture
- Specific to certain types of information processing, such as e-Commerce or supply chain management
- Specific to certain vertical industries; for example, the models generated by vertical consortia such as TMF (in the Telecommunications sector), ARTS (Retail) or POSC (Petrotechnical)

Enterprise Architect's support for the Enterprise Continuum is provided by the Enterprise Continuum diagram and the corresponding Diagram Toolbox page. The starter model consists of an interface to the TOGAF Enterprise Continuum.

When you double-click on an Architecture Continuum or Solution Continuum element, an Enterprise Continuum diagram displays. The Diagram Toolbox page provides the Architecture Building Block and Solutions Building Block elements and the appropriate relationship connectors.
Support For Federal Enterprise Architecture Framework

TOGAF provides diagrams and Toolbox pages specific to the Federal Enterprise Architecture Framework (FEAF). It also provides 'out-of-the-box' models of the FEAF Performance Reference model and Technical Reference model.

To open FEAF-PRM and FEAF-TRM models:

1. Create a new Enterprise Architect project file, and click on the top-level Package.
2. Select the 'Design > Model > Perspectives > Model Wizard' option.
3. In the Model Wizard, select the 'Enterprise Architecture > TOGAF' Perspective and the required FEAF Pattern.
4. Click on the Create Patterns button.

These TOGAF Toolbox pages provide specific support for FEAF:

- FEAF Business Reference Model Toolbox Page
- FEAF Performance Reference Model Toolbox Page
- FEAF Service Component Reference Model Toolbox Page
- FEAF Technical Reference Model Toolbox Page
TOGAF Catalogs

Enterprise Architect helps you to create Model Catalog Artifacts, using the TOGAF-Catalog model Pattern. Choosing this model Pattern in the Model Wizard generates a template model in which you create TOGAF-specific catalogs for:

- Actors
- Business Services
- Organization Units
- Principles
- Requirements and
- Roles

Each Model View will list all objects of the corresponding type in the entire model.

Alternatively, you can create TOGAF Catalogs in a diagram using Model View elements from the 'Dashboard' pages of the Diagram Toolbox.
UPDM

The MDG Technology for UPDM (Unified Profile for DoDAF-MODAF) provides a UML profile that extends the capability of Enterprise Architect to provide a standard approach for modeling systems and Enterprise Architectures in support of DoDAF and MODAF.

DoDAF is the abbreviation of Department of Defense Architecture Framework (USA); MODAF is the abbreviation of Ministry of Defence Architecture Framework (UK).

Discussion

The topics described here provide an introduction to, and procedural explanation of, using the MDG Technology for UPDM in Enterprise Architect.

<table>
<thead>
<tr>
<th>Section</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>Welcome</td>
<td>This section provides an introduction to UPDM, and contains the formal documentation defining its use with Enterprise Architect.</td>
</tr>
<tr>
<td>Using UPDM</td>
<td>Get started with UPDM, learning about the model structure, templates, diagram types and more.</td>
</tr>
<tr>
<td>Model Validation</td>
<td>Learn how to develop and configure model validation for UPDM.</td>
</tr>
</tbody>
</table>
Welcome to the MDG Technology for UPDM

Welcome to the MDG Technology for UPDM 2.0, in Sparx Systems Enterprise Architect.

This Technology provides a UML profile that extends the capability of Enterprise Architect to support the creation of Unified Profile for DoDAF and MODAF (UPDM) architecture models. DoDAF is the abbreviation of Department of Defense Architecture Framework (USA); MODAF is the abbreviation of Ministry of Defence Architecture Framework (UK).

The UPDM profile provides a standard approach for modeling systems and enterprise architectures in support of DoDAF and MODAF. It improves interoperability of architecture data among architecture modeling tools, enhances reuse of architecture data, and improves communication among DoDAF and MODAF stakeholders.

The Technology is already integrated with the Enterprise Architect Ultimate and Unified Editions; it can be purchased separately to be used with the Enterprise Architect Professional or Corporate Editions.

This technology is based on the Unified Profile for DoDAF-MODAF (UPDM) version 1.0. UPDM 1.0 is based on DoDAF version 1.5 and MODAF version 1.2. Full details of the profile, including the latest specification, can be obtained from the Object Management Group (OMG) website.

Getting Started

For instructions on how to use the MDG Technology for UPDM, see the topics Getting Started with MDG Technology for UPDM and Using UPDM.
MDG Technology for UPDM Copyright Notice

Copyright © 2010 - 2020 Sparx Systems Pty. Ltd. All rights reserved.

The software contains proprietary information of Sparx Systems Pty Ltd. It is provided under a license agreement containing restrictions on use and disclosure and is also protected by copyright law. Reverse engineering of the software is prohibited. Please read the product license agreement for full details.

Due to continued product development, this information may change without notice. The information and intellectual property contained herein is confidential between Sparx Systems and the client and remains the exclusive property of Sparx Systems. If you find any problems in the documentation, please report them to us in writing. Sparx Systems does not warrant that this document is error-free. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise without the prior written permission of Sparx Systems. Licensed users are granted the right to print a single hardcopy of the user manual per licensed copy of the software, but may not sell, distribute or otherwise dispose of the hardcopy without written consent of Sparx Systems.

Sparx Systems Pty. Ltd.
99 Albert St,
Creswick, Victoria 3363,
AUSTRALIA
Phone: +61 (3) 5345 1140
Fax: +61 (3) 5345 1104
Support Email: support@sparxsystems.com
Sales Email: sales@sparxsystems.com
Website: sparxsystems.com
MDG Technology for UPDM Software Product License Agreement

This Software Product License Agreement relates to the separately-purchased MDG Technology for UPDM for use with the Corporate and Professional Editions of Sparx Systems Enterprise Architect. Where the MDG Technology for UPDM is integrated with the Ultimate and Unified Editions of Enterprise Architect, this is covered by the Sparx Systems Enterprise Architect Modelling Tool.

MDG Technology for UPDM, Enterprise Architect MDG Add-In, Version 2.0.

Copyright (C) 2010 - 2020 Sparx Systems Pty Ltd. All Rights Reserved

IMPORTANT-READ CAREFULLY: This End User License Agreement ("EULA") is a legal agreement between YOU as Licensee and SPARX for the SOFTWARE PRODUCT identified above. By installing, copying, or otherwise using the SOFTWARE PRODUCT, YOU agree to be bound by the terms of this EULA. If YOU do not agree to the terms of this EULA, promptly delete the unused SOFTWARE PRODUCT.

The copyright in the SOFTWARE PRODUCT and its documentation is owned by Sparx Systems Pty Ltd, A.B.N 38 085 034 546. Subject to the terms of this EULA, YOU are granted a non-exclusive right for the duration of the EULA to use the SOFTWARE PRODUCT. YOU do not acquire ownership of copyright or other intellectual property rights in any part of the SOFTWARE PRODUCT by virtue of this EULA.

Your use of this software indicates your acceptance of this EULA and warranty.

DEFINITIONS

In this End User License Agreement, unless the contrary intention appears:

- "EULA" means this End User License Agreement
- "SPARX" means Sparx Systems Pty Ltd A.C.N 085 034 546
- "LICENSEE" means YOU, or the organization (if any) on whose behalf YOU are taking the EULA
- "Registered Edition of MDG Technology for UPDM" means the edition of the SOFTWARE PRODUCT, which is available for purchase from the web site: https://sparxsystems.com/updm/purchase.html, following a thirty-day free evaluation period
- "SOFTWARE PRODUCT" or "SOFTWARE" means MDG Technology for UPDM, which includes computer software and associated media and printed materials, and may include online or electronic documentation
- "SUPPORT SERVICES" means email-based support provided by SPARX, including advice on usage of the SOFTWARE PRODUCT, investigation of bugs, fixes, repairs of models, if and when appropriate, and general product support
- "SPARX SUPPORT ENGINEERS" means employees of SPARX who provide on-line support services
- "TRIAL EDITION" means the edition of the SOFTWARE PRODUCT, which is available free of charge for evaluation purposes for a period of thirty (30) days

GRANT OF LICENSE

In accordance with the terms of this EULA, YOU are granted the following rights:

- To install and use one copy of the SOFTWARE PRODUCT, or in its place, any prior version for the same operating system, on a single computer; as the primary user of the computer on which the SOFTWARE PRODUCT is installed, YOU may make a second copy for your exclusive use on either a home or portable computer
- To store or install a copy of the SOFTWARE PRODUCT on a storage device, such as a network server, used only to install or run the SOFTWARE PRODUCT over an internal network; if YOU wish to increase the number of users entitled to concurrently access the SOFTWARE PRODUCT, YOU must notify SPARX and agree to pay an
additional fee

- To make copies of the SOFTWARE PRODUCT for backup and archival purposes only

EVALUATION LICENSE

The TRIAL EDITION is not free software. Subject to the terms of this agreement, YOU are hereby licensed to use the SOFTWARE PRODUCT for evaluation purposes without charge for a period of thirty (30) days.

Upon expiration of the thirty (30) days, the Software Product must be removed from the computer. Unregistered use of the SOFTWARE PRODUCT after the 30-day evaluation period is in violation of Australian, U.S. and international copyright laws.

SPARX may extend the evaluation period on request and at their discretion.

If YOU choose to use the SOFTWARE PRODUCT after the 30-day evaluation period, a license must be purchased (as described at https://sparxsystems.com/updn/purchase.html). Upon payment of the license fee, YOU will be sent details on where to download the registered edition of the software product and will be provided with a suitable software ‘key’ by email.

ADDITIONAL RIGHTS AND LIMITATIONS

YOU hereby undertake not to sell, rent, lease, translate, adapt, vary, modify, decompile, disassemble, reverse engineer, create derivative works of, modify, sub-license, loan or distribute the SOFTWARE PRODUCT other than as expressly authorized by this EULA.

YOU further undertake not to reproduce or distribute license key-codes except under the express and written permission of SPARX.

If the Software Product purchased is an ACADEMIC EDITION, YOU acknowledge that the license is limited to use in an educational context, either for self-education or use in a registered teaching institution. The ACADEMIC EDITION may not be used to produce commercial software products or be used in a commercial environment, without the express written permission of SPARX.

ASSIGNMENT

YOU may only assign all your rights and obligations under this EULA to another party if YOU supply to the transferee a copy of this EULA and all other documentation including proof of ownership. Your license is then terminated.

TERMINATION

Without prejudice to any other rights, SPARX may terminate this EULA if YOU fail to comply with the terms and conditions. Upon termination YOU or YOUR representative shall destroy all copies of the SOFTWARE PRODUCT and all of its component parts or otherwise return or dispose of such material in the manner directed by SPARX.

WARRANTIES AND LIABILITY

WARRANTIES

SPARX warrants that:

- The SOFTWARE PRODUCT will perform substantially in accordance with the accompanying written materials for a period of ninety (90) days from the date of receipt, and
• Any SUPPORT SERVICES provided by SPARX shall be substantially as described in applicable written materials provided to YOU by SPARX, and SPARX SUPPORT ENGINEERS will make commercially reasonable efforts to solve any problems associated with the SOFTWARE PRODUCT.

EXCLUSIONS
To the maximum extent permitted by law, SPARX excludes, for itself and for any supplier of software incorporated in the SOFTWARE PRODUCT, all liability for all claims, expenses, losses, damages and costs made against or incurred or suffered by YOU directly or indirectly (including without limitation lost costs, profits and data) arising out of:

• YOUR use or misuse of the SOFTWARE PRODUCT;
• YOUR inability to use or obtain access to the SOFTWARE PRODUCT;
• Negligence of SPARX or its employees, contractors or agents, or of any supplier of software incorporated in the SOFTWARE PRODUCT, in connection with the performance of SPARX's obligations under this EULA; or
• Termination of this EULA by either party for any reason.

LIMITATION
The SOFTWARE PRODUCT and any documentation are provided "AS IS" and all warranties, whether express, implied, statutory or otherwise, relating in any way to the subject matter of this EULA or to this EULA generally, including without limitation, warranties as to: quality; fitness; merchantability; correctness; accuracy; reliability; correspondence with any description or sample, meeting your or any other requirements; uninterrupted use; compliance with any relevant legislation; and being error or virus free are excluded. Where any legislation implies in this EULA any term, and that legislation avoids or prohibits provisions in a contract excluding or modifying such a term, such term shall be deemed to be included in this EULA. However, the liability of SPARX for any breach of such term shall, if permitted by legislation, be limited, at SPARX's option to any one or more of the following upon return of the SOFTWARE PRODUCT and a copy of the receipt:

• If the breach relates to the SOFTWARE PRODUCT:
  • The replacement of the SOFTWARE PRODUCT, or the supply of an equivalent SOFTWARE PRODUCT;
  • The repair of such SOFTWARE PRODUCT, or the payment of the cost of replacing the SOFTWARE PRODUCT, or of acquiring an equivalent SOFTWARE PRODUCT; or
  • The payment of the cost of having the SOFTWARE PRODUCT repaired.
• If the breach relates to services in relation to the SOFTWARE PRODUCT:
  • The supplying of the services again; or
  • The payment of the cost of having the services supplied again.

TRADEMARKS
All names of products and companies used in this EULA, the SOFTWARE PRODUCT, or the enclosed documentation may be trademarks of their corresponding owners. Their use in this EULA is intended to be in compliance with the respective guidelines and Licenses. Windows, Windows XP, Windows 2003 Server, Windows 2008 Server, Windows Vista, Windows 7 and Windows 8 are trademarks of Microsoft®.

GOVERNING LAW
This agreement shall be construed in accordance with the laws of the Commonwealth of AUSTRALIA, in the state of Victoria.
Acknowledgement of Trademarks - UPDM

Trademarks of Microsoft

- Microsoft®
- Windows®

Trademarks of the OMG

- OMG™
- Object Management Group™
- UML™
- Unified Modeling Language™
MDG Technology for UPDM Support

Technical support for the MDG Technology for UPDM is available to registered users of Enterprise Architect in exactly the same way as for Enterprise Architect itself.
MDG Technology for UPDM System Requirements

The MDG Technology for UPDM version 2.0 runs under these environments:

Operating Systems

- Windows 8
- Windows 7
- Windows Vista
- Windows 2008 Server
- Windows 2003 Server
- Windows XP Service Pack 2

Enterprise Architect Versions

- Enterprise Architect Version 9.0 or later
Getting Started with MDG Technology for UPDM

When you install the Unified or Ultimate Edition of Enterprise Architect, the MDG Technology for UPDM is fully enabled and ready to use.

If you have the Corporate or Professional Edition of Enterprise Architect, you can purchase and install the MDG Technology for UPDM separately; once you have entered the registration key for the MDG Technology for UPDM, it is automatically available in and integrated with Enterprise Architect, as for the Unified and Ultimate Editions.

Access the MDG Technology

1. Create a new Enterprise Architect project file, and click on the top-level Package.
2. Select the 'Design > Model > Perspectives > Model Wizard' option.
3. In the Model Wizard, select the 'Enterprise Architecture > UPDM' Perspective and the 'UPDM Frameworks' Pattern Group; select either the 'DoDAF Framework' Pattern or the 'MODAF Framework' Pattern.
4. Click on the Create Patterns button.

A new base DoDAF or MODAF model is created in the Browser window.
Using UPDM

UPDM is the Unified Profile for the Department of Defense Architecture Framework (DoDAF) and Ministry of Defence Architecture Framework (MODAF). UPDM is an Object Management Group (OMG) initiative; the specification is available from the OMG website.

You can use the MDG Technology for UPDM to perform UPDM modeling within Enterprise Architect. The Technology provides:

- The UPDM Profile, which defines the stereotyped UML elements that are used for UPDM modeling
- Custom diagram types for each UPDM view
- Custom Diagram Toolbox pages for each UPDM diagram type, which give easy access to the elements used on diagrams of that type
- Options within the Model Wizard that can be used to import a template Package for each UPDM view and that provide a brief description of the view and what might be expected of the modeler
- Quicklinks for stereotyped elements that guide you towards creating correct relationships between elements
- Model Validation rules that you can apply to check your models for correctness
- Relationship Matrix profiles for showing the relationships between elements
- Model Views that help you navigate your model quickly to find specific diagram more easily
- A Glossary import, with items describing each UPDM stereotype for easy reference
- Tagged Values that you can use to enter metadata specific to UPDM elements
- An Example Model that illustrates a typical UPDM problem and its solution, implemented using Enterprise Architect
Model Wizard in the MDG Technology for UPDM

You can create UPDM models within your project using templates selected from the Enterprise Architect Model Wizard

Access

<table>
<thead>
<tr>
<th>Ribbon</th>
<th>Design &gt; Model &gt; Add &gt; Model Wizard &gt; Model Patterns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Context Menu</td>
<td>Browser window</td>
</tr>
<tr>
<td>Keyboard Shortcuts</td>
<td>Ctrl+Shift+M</td>
</tr>
</tbody>
</table>

Notes

- In the Model Wizard, click on the '... Perspective' button and select 'System Engineering > UPDM'
- Expand the 'UPDM Frameworks' group or one of the 'DoDAF' or 'MODAF' groups, and click on the required Pattern in that group
- Click on the Create Patterns button to generate the corresponding UPDM model structures in your project
UPDM Framework Diagram

When developing and distributing a model, it is useful to have a single front page diagram that has hyperlinks to all the important information in the model. That is the aim of the two UPDM Framework diagrams (one for DoDAF, one for MODAF), which are created with color-coded swimlanes for each set of views. You can drag and drop on to these diagrams:

- Packages, which act as hyperlinks to the documents that they own
- Document Artifacts
- Any kind of composite element that points to its child diagram
- Hyperlinks pointing to custom SQL queries, Relationship Matrix profiles or external files

Create a UPDM Framework Diagram

1. In the Model Wizard, click on the '... Perspective' button and select 'System Engineering > UPDM'.
2. Expand the 'UPDM Frameworks' group and click on the required Pattern, either 'DODAF Framework' or 'MODAF Framework'.
3. Click on the Create Patterns button to generate the corresponding UPDM model structures in your project.

Editing Swimlanes

You can add, remove and modify the swimlanes on the Framework diagram. Select 'Design > Diagram > Manage > Swimlanes'.

To change the width of swimlanes, use the mouse to drag their boundaries.
Changing Appearances

Each Package, document and hyperlink on the Framework diagram has an alternative image. To load these images into your model, select the Configure > Reference Data > Images' option.

If you want to apply your own bitmap images to the UPDM elements, you must first import the images into the model, also using the 'Configure > Reference Data > Images' option. Then you can either select the element and press Ctrl+Shift+W to add an alternative image to the element, or you can apply your own stereotype to apply a Shape Script to the element. For example, you might define a stereotype with this Shape Script:

```plaintext
shape main
{
    v_align="center";
    h_align="center";
    defSize(90,70);
    image("myBitMap.bmp",0,0,100,100);
    printWrapped("#name#");
}
```
UPDM Extensions Menu

You can perform various tasks on your UPDM model using the UPDM Technology menu.

Access

<table>
<thead>
<tr>
<th>Ribbon</th>
<th>Specialize &gt; Technologies &gt; UPDM 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Context Menu</td>
<td>Right-click on Package, diagram or element</td>
</tr>
</tbody>
</table>

Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synchronize Tagged Values</td>
<td>Add missing Tagged Values to all elements in the model that require them.</td>
</tr>
<tr>
<td>Import Glossary</td>
<td>Import UPDM information into the Enterprise Architect Glossary.</td>
</tr>
<tr>
<td>Import Images</td>
<td>Import the alternative images (as used in the UPDM Framework diagram) into the current model. You can use these images to decorate your own models (select a diagram object, right-click and select 'Appearance</td>
</tr>
<tr>
<td>Help</td>
<td>Display this Help topic.</td>
</tr>
<tr>
<td>About</td>
<td>Show the version of the MDG Technology for UPDM that you are using. The version number has the format 1.0.001, where 1.0 is the version of the UPDM specification that is supported, and 001 is the incremental build number.</td>
</tr>
</tbody>
</table>
UPDM Diagram Types

The MDG Technology for UPDM introduces a number of custom diagram types into Enterprise Architect. These are, for the most part, extended UML diagrams. On opening a UPDM diagram, Enterprise Architect automatically opens the appropriate UPDM Diagram Toolbox pages for the diagram type.

You can use the UPDM diagrams that are generated by the Model Wizard, or create a new UPDM diagram.

Access

<table>
<thead>
<tr>
<th>Ribbon</th>
<th>Design &gt; Diagram &gt; Add</th>
</tr>
</thead>
<tbody>
<tr>
<td>Context Menu</td>
<td>Browser window</td>
</tr>
</tbody>
</table>

Notes

- On the 'New Diagram' dialog, select 'UPDM' in the 'Select From' panel and the appropriate diagram type in the 'Diagram Types' panel
- Click on the OK button to open the Diagram View with the empty diagram displayed
UPDM Toolboxes

When you open a diagram, Enterprise Architect opens the Diagram Toolbox pages that are most useful for that particular diagram type. For a UPDM diagram, Enterprise Architect opens the Toolbox pages that contain elements and relationships appropriate to the particular View that the diagram is part of, as well the pages for the diagram type. For example, if you open an SOV-5 Activity diagram, Enterprise Architect opens the 'SOV-5 Elements' page, the 'UML Activity' page and the 'UML Activity Relationships' page.

In addition, the 'Common' elements and relationships page and the 'Artifacts' page of the Diagram Toolbox are always available, regardless of which diagram is open.

If you hide the default Toolboxes and want to get them back, simply switch to the Start Page and back to the current
diagram, and all the default Toolboxes for the current diagram type are re-opened.

**All UPDM Stereotypes**

For your convenience, a Diagram Toolbox page is provided that includes every stereotype in the UPDM profile, listed in alphabetical order. If you cannot remember which context-sensitive Toolbox page a stereotype appears in, just go to the 'All UPDM Stereotypes' Toolbox page instead. To make this page available at all times, either:

- Select the 'Specialize > Technologies > Manage-Tech' ribbon option, select 'UPDM Technology' in the table, and click on the Set Active button, or
- Select 'UPDM 2.0' from the list box on the Default Tools toolbar
UPDM Stereotypes

**ActualMeasurementSet**

A set or collection of measurements; used in AV-3, OV-3, SV-6 and SV-7.

Extensions:
- Object

Constraints:
- Classifier must be a MeasurementSet

Use:
- Press Ctrl and drag a MeasurementSet element from the Browser window to create an instance, or drop an ActualMeasurementSet from the Diagram Toolbox and press Ctrl+L to set the classifier; set the Run State and enter actual values for each of the classifier's attributes

**ActualOrganization**

An actual specific organization as an instance of an organization Class; used in AcV-1, OV-4, StV-5, TV-1 and TV-2.

Extensions:
- Object

Generalizations:
- ActualOrganizationalResource

Constraints:
- Classifier must be an Organization

Use:
- Press Ctrl and drag an Organization from the Browser window to create an instance, or drop an ActualOrganization from the Diagram Toolbox and press Ctrl+L to set the classifier
- Can have a set of 'ratifiedStandards' (Standard)
- Can be 'responsibleFor' a set of ActualProject
- Can be client and/or supplier of an ActualOrganizationRelationship
- Can be client of an OwnsProcess dependency to an OperationalActivity

**ActualOrganizationRelationship**

A relationship between two actual organizational resources (organizations or posts); used in OV-4.

Extensions:
- InformationFlow

Constraints:
- Supplier must be an ActualOrganizationalResource (ActualOrganization or ActualPost)
- Client must be an ActualOrganizationalResource (ActualOrganization or ActualPost)
- Realizes a ResourceInteraction
ActualPerson

A named individual that fulfills an ActualPost; used in OV-4.

Extensions:
- Object

Constraints:
- Classifier must be a Person

Use:
- Press Ctrl and drag a Person from the Browser window to create an instance, or drop an ActualPerson from the Diagram Toolbox and press Ctrl+L to set the classifier
- Can be a client of a FillsPost dependency to an ActualPost

ActualPost

An actual, specific post, as an instance of the Post Class; used in AcV-1, OV-4 and StV-5.

Extensions:
- Object

Generalizations:
- ActualOrganizationalResource

Constraints:
- Classifier must be a Post

Use:
- Press Ctrl and drag a Post from the Browser window to create an instance, or drop an ActualPost from the Diagram Toolbox and press Ctrl+L to set the classifier
- Can be responsible for a set of ActualProject
- Can be client and/or supplier of an ActualOrganizationRelationship
- Can be client of an OwnsProcess dependency to an OperationalActivity
- Can be supplier of a FillsPost dependency from an ActualPerson

ActualProject

A time-limited attempt to create a specific set of products or services; used in AcV-1, AcV-2, StV-3, StV-5 and SV-8.

Extensions:
- Object

Constraints:
- Classifier must be a Project

Use:
- Press Ctrl and drag a Project from the Browser window to create an instance, or drop an ActualProject from the Diagram Toolbox and press Ctrl+L to set the classifier
- Can have Aggregations to or from another ActualProject
- Can have a set of ‘ownedMilestones’ (type ActualProjectMilestone, including IncrementMilestone, OutOfServiceMilestone, NoLongerUsedMilestone and DeployedMilestone)
**ActualProjectMilestone**

An event in a project by which progress is measured; used in AcV-2, StV-3, StV-5 and SV-8.

See also: IncrementMilestone, OutOfServiceMilestone, NoLongerUsedMilestone and DeployedMilestone.

Extensions:
- Object

Specializations:
- IncrementMilestone
- OutOfServiceMilestone
- NoLongerUsedMilestone
- DeployedMilestone

Constraints:
- Classifier must be a ProjectMilestone

Use:
- Can have a set of associated Resource
- Can be client/supplier of a MilestoneSequence

**Alias**

A comment used to define an alternative name for an element; used in AV-2.

Extensions:
- Note

Constraints:
- AnnotatedElement must be a UPDElement

Use:
- Just drag a Quicklink NoteLink from the Alias to the annotated element

**Arbitrary Relationship**

Represents a visual indication of a connection used in high level operational concept diagrams. The connections are purely visual and cannot be related to any architectural semantics; used in OV-1.

Extensions:
- Dependency

Constraints:
- Client and Supplier must both be stereotyped ConceptRole

Use:
- Drag a Quicklink from a ConceptRole

**ArchitecturalDescription**

A specification of a system of systems at a technical level, which also provides the business context; used in AV-1.
Extensions:
- Package

Use:
- Can have a DefinesArchitecture Realization to an EnterprisePhase
- Can have an ArchitecturalReference Dependency to another ArchitecturalDescription
- Can be annotated with an ArchitectureMetadata note

**ArchitecturalReference**

Asserts that one architectural description refers to another; used in AV-1.

Extensions:
- Dependency

Constraints:
- Client and Supplier must both be stereotyped ArchitecturalDescription

Use:
- Drag a Quicklink from an ArchitecturalDescription.

**ArchitectureMetadata**

Information on architectural description; used in AV-1.

Extensions:
- Note

Generalizations:
- Metadata

Constraints:
- AnnotatedElement must be an ArchitecturalDescription

Use:
- Drag a quicklink from an ArchitecturalDescription

**Capability**

A high-level specification of the enterprise's ability; used in AV-1, OV-2, SOV-3, StV-1, StV-2, StV-3, StV-4, StV-5, StV-6, SV-1 and SV-9.

Extensions:
- Class

Generalizations:
- SubjectOfForecast

Use:
- Can have a set of associated environment conditions stereotyped Environment
- Capabilities can be composed of Capabilities (Composite aggregation)
- Capabilities can be dependent on Capabilities (Dependency)
• Capabilities can sub-class Capabilities (Generalization)
• Can be supplier or client of a Forecast (both must be same stereotype) (from SubjectOfForecast)

CapabilityConfiguration

A set of physical and human resources (and their interactions) configured to provide a capability; used in OV-1, OV-2, OV-3, StV-3, StV-5, SV-1, SV-3, SV-9, SV-10a, SV-12, TV-1, TV-2 and AcV-2.

Extensions:
• Class
Generalizations:
• Resource
• ConceptItem
• Performer
• ResourceInteractionItem
• SubjectOfResourceConstraint
• SubjectOfForecast
• SystemsElement
• SubjectOfResourceStateMachine
• ResourceInteractionItem
Specializations:
• SystemsNode
Use: Can:
• Have a set of associated deployed milestones, stereotyped DeployedMilestone
• Have an optional associated no longer used milestone, stereotyped No Longer Used Milestone
• Have a set of associated increment milestones, stereotyped IncrementMilestone
• Have an optional associated out of service milestone, stereotyped Out Of Service Milestone
• Be annotated by a StandardConfiguration note
• Be the type of a ConceptRole (from ConceptItem)
• Have a set of associated milestones, stereotyped ActualProjectMilestone (from Resource)
• Be the client of a RealizesCapability Realization to a Capability (from Resource)
• Be the client of a ProvidesCompetence Dependency to a Competence (from Resource)
• Have an attached ResourceConstraint (from Resource, SubjectOfResourceConstraint)
• Be the supplier or client of a Forecast Dependency (both must have same stereotype) (from SubjectOfForecast)
• Own a ServicePoint (from Resource)
• Own a RequestPoint (from Resource)
• Own a ResourcePort (from Resource)
• Be the source and target of a ResourceInteraction (from Resource)
• Own a ServiceOperation (from Resource)
• Be the type of a KnownResource (from Resource)
• Be the type of a ResourceRole (from Resource)
• Have a Performs Dependency to a PerformedActivity (Function or OperationalActivity) (from Performer)
Climate

A type of weather condition, or combination of weather conditions, in which a Performer performs; used in StV-2.
Extensions:
• Class
Generalizations:
• EnvironmentalType
Use:
• Can be the type of an EnvironmentProperty

Commands

Asserts that one OrganizationalResource commands another; used in OV-4, SV-1 and SV-10c.
Extensions:
• InformationFlow
Generalizations:
• ResourceInteraction
Constraints:
• Source must be an OrganizationalResource
• Target must be an OrganizationalResource
Use:
• Conveys a DataElement

CompatibleWith

Relates a node to a location to assert that the operational node must be situated at that location; used in OV-2.
Extensions:
• Dependency
Constraints:
• Client is a Node
• Supplier is a ReferredLocation (Location or PhysicalLocation)
Use:
• Drag a Quicklink from a Node

Competence

A specific set of abilities defined by knowledge, skills and attitude; used in OV-4, SV-1 and SV-9.
Extensions:
• Class
Generalizations:
- **SubjectOfForecast**
  Use: Can be:
  - The supplier or client of a Forecast Dependency (both must have same stereotype) (from SubjectOfForecast)
  - The supplier of a ProvidesCompetence Dependency
  - The supplier of a RequiresCompetence Dependency

**ConceptRole**

A relationship that asserts that a ConceptItem forms part of the high level operational concept; used in OV-1.

Extensions:
- **Part**

Constraints:
- Type is a ConceptItem

Use:
- Owned by a HighLevelOperationalConcept
- Can be supplier and client of an ArbitraryRelationship dependency

**ConfigurationExchange**

CapabilityConfigurations that are exchanged between Nodes; used in OV-2, OV-3 and OV-6c.

Extensions:
- **InformationFlow**

Generalizations:
- **OperationalExchange**

Constraints:
- Source is a Node (from OperationalExchange)
- Target is a Node (from OperationalExchange)

Use:
- Conveys a CapabilityConfiguration

**Controls**

A type of ResourceInteraction where one Resource controls another; used in SV-1 and SV-10c.

Extensions:
- **InformationFlow**

Generalizations:
- **ResourceInteraction**

Constraints:
- Source is an OrganizationalResource (Organization or Post)
- Target is a ManufacturedResourceType (ResourceArtifact or Software)

Use:
• Conveys a DataElement

DataElement

A formalized representation of data that is managed by or exchanged between systems; used in OV-4, SV-1, SV-2, SV-4, SV-6, SV-10a, SV-10b and SV-11.

Extensions:
• Class
Generalizations:
• SubjectOfResourceConstraint
• ResourceInteractionItem
• SystemsElement
• SubjectOfResourceStateMachine

Use:
• Can have an attached ResourceConstraint (from SubjectOfResourceConstraint)
• Can have a set of associated defined EntityItems
• Can be conveyed on a Controls or Commands information flow

DataExchange

A DoDAF alias for ResourceInteraction.

Extensions:
• InformationFlow
Generalizations:
• ResourceInteraction
• SystemsElement

Use:
• Conveys ResourceInteractionItem (Energy, Post, Organization, CapabilityConfiguration, Software, ResourceArtifact, or DataElement)

DefinesArchitecture

Establishes a relationship between ArchitecturalDescription and EnterprisePhase; used in AV-1.

Extensions:
• Realization

Constraints:
• Client is an ArchitecturalDescription
• Supplier is an EnterprisePhase

Use:
• Drag a Quicklink from an ArchitecturalDescription
Definition

A definition of an element in the architecture; used in AV-2.

Extensions:
- Note

Constraints:
- Annotated Element is a UPDMElement

Use:
- Drop from toolbox and drag a NoteLink to any UPDM element

DeployedMilestone

Asserts that an ActualOrganizationResource started to use, or is slated to start using, a CapabilityConfiguration from a specific point in time; used in StV-5.

Extensions:
- Object

Generalizations:
- ActualProjectMilestone

Constraints:
- Classifier must be a ProjectMilestone (from ActualProjectMilestone)

Use: Can:
- Have a set of associated (usedBy) ActualOrganizationResource (ActualOrganization or ActualPost)
- Have a set of associated Resource (from ActualProjectMilestone)
- Be client/supplier of a MilestoneSequence (from ActualProjectMilestone)

EnduringTask

A type of behavior recognized by an enterprise as being essential to achieving its goals - that is, a strategic specification of what the enterprise does; used in StV-1.

Extensions:
- Class

Use:
- Target of association from EnterprisePhase

Energy

Energy to be exchanged between Nodes; used in OV-2, OV-3, OV-5, SV-1, SV-4 and SV-6.

Extensions:
- Class

Generalizations:
- ResourceInteractionItem
- OperationalExchangeItem
Use:
- Conveyed on an EnergyExchange information flow

**EnergyExchange**

A relationship specifying the need to exchange energy between nodes; used in OV-2, OV-3 and OV-6c.

Extensions:
- InformationFlow

Generalizations:
- OperationalExchange
- OperationalElement

Constraints:
- Source is a Node (from OperationalExchange)
- Target is a Node (from OperationalExchange)

Use:
- Conveys a Class stereotyped Energy

**EnterpriseGoal**

A specific required objective of the enterprise that the architecture represents; used in StV-1.

Extensions:
- Class

Use:
- Has an association to one EnterprisePhase

**EnterprisePhase**

A specific, required objective of the enterprise that the architecture represents; used in AV-1, StV-1, StV-2, StV-5 and SV-9.

Extensions:
- Class

Specializations:
- WholeLifeEnterprise

Use:
- Can have a set of associations (statementTasks) to EnduringTask Class
- Can have a set of associations (exhibits) to Capability Class
- Can have a set of associations (inhabits) to Environment Class
- Can have a set of associations (goals) with EnterpriseGoal Class
- Can have a set of associations (visions) with EnterpriseVision Class
- Can be the type of a StructuralPart or TemporalPart
- Fulfills a Mission Use Case
• Can be Supplier of a DefinesArchitecture Realization

**EnterpriseVision**

The overall aims of an enterprise over a given period of time; used in StV-1.

Extensions:
• Class
Use:
• Has an association to one EnterprisePhase

**EntityAttribute**

A defined property of an EntityItem; used in OV-7 and SV-11.

Extensions:
• Attribute
Use:
• Is owned by an EntityItem

**EntityItem**

A definition (type) of an item of interest; used in OV-7 and SV-11.

Extensions:
• Class
Constraints:
• Owned attributes must be stereotyped EntityAttribute
Use: Can:
• Be owned by a DataModel
• Be the end type of an EntityRelationship
• Have a set of associated (definedBy) DataElement
• Have a set of associated (represents) InformationElement
• Be conveyed on a Commands or Controls information flow

**EntityRelationship**

Asserts that there is a relationship between two EntityItems; used in OV-7 and SV-11.

Extensions:
• Association
Constraints:
• The types of any object at either end must be stereotyped EntityItem
Environment

A definition of the conditions in which the Enterprise exists or functions; used in AV-1 and StV-2.

Extensions:
- Class

Constraints:
- Owned attributes must be EnvironmentProperty

EnvironmentProperty

Asserts that an Environment has one or more properties such as Climate, Location or LightCondition; used in StV-2.

Extensions:
- Attribute

Constraints:
- Type must be an EnvironmentalType (LightCondition, Location, PhysicalLocation or Climate)

Use:
- Owned by an Environment element

Equipment

A physical resource that is used to accomplish a task or function in a system or an environment; used in SV-1.

Extensions:
- Part

Generalizations:
- ResourceRole

Constraints:
- Class must be an OrganizationResource (Organization or Post)
- Type must be a ResourceArtifact

Use:
- Can have a RequiresCompetence dependency to a Competence (from ResourceRole)
- Can have a set of associations (usedFunctions) to Function (from ResourceRole)

ExhibitsCapability

Assertion that a Node is required to have a Capability; used in OV-2.

Extensions:
- Dependency

Constraints:
- Client must be a Node
- Supplier must be a Capability
Expose

Assertion that a service interface exposes a capability.
Extensions:
- Dependency
Constraints:
- Client must be a ServiceInterface
- Supplier must be a Capability

ExternalIndividual

An individual defined by an external ontology; used in AV-2.
Extensions:
- Object
Use:
- Can be the supplier of a SameAs dependency

ExternalNode

Operational node that is external to the architecture; used in OV-2.
Extensions:
- Class
Generalizations:
- Node
- Performer
Use: Can:
- Own a RequestPoint Port (from Node)
- Own a ServicePoint Port (from Node)
- Be client of an ExhibitsCapability dependency to a Capability (from Node)
- Have a Performs dependency to a PerformedActivity (Function or OperationalActivity) (from Performer)
- Have a CompatibleWith dependency to a ReferredLocation (PhysicalLocation or Location) (from Node)

ExternalType

A type defined by an external ontology; used in AV-2.
Extensions:
- Class
Use:
- Can be the Supplier of a SameAs dependency
- Any UPDM element can have a Generalization to an ExternalType
**FieldedCapability**

A deployed and fully realized instance of a capability; used in SV-2.

Extensions:
- Object

Constraints:
- Its classifier must be a CapabilityConfiguration

**FillsPost**

Asserts that ActualPerson fills an ActualPost; used in OV-4.

Extensions:
- Dependency

Constraints:
- Client must be an ActualPerson
- Supplier must be an ActualPost

**Forecast**

The actual or predicted status of a system at a project milestone; used in SV-9.

Extensions:
- Dependency

Specializations:
- TechnologyForecast

Constraints:
- Client and Supplier are both SubjectOfForecast (Standard, Competence, Capability, CapabilityConfiguration, Organization, Post, ResourceArtifact or Software)
- Client and Supplier must be the same specialization of SubjectOfForecast

**Function**

An activity that is specified in context of the resource that performs it; used in OV-4, SV-1, SV-4, SV-5 and SV-10a.

Extensions:
- Activity

Generalizations:
- PerformedActivity
- SystemsElement
- SubjectOfResourceConstraint

Constraints:
- Owned parameters are FunctionParameter
Use: Can:
- Be Supplier of a Performs dependency (from PerformedActivity)
- Own ServiceOperationAction, FunctionAction and FunctionEdge
- Be Client of an ImplementsOperational dependency to an OperationalActivity (from SystemsElement)
- Have an attached ResourceConstraint (from SubjectOfResourceConstraint)

**FunctionAction**

A call behavior action that invokes the function that needs to be performed; used in SV-4.

Extensions:
- Action (Call Behavior)

Specializations:
- SystemFunctionAction

Constraints:
- Activity is stereotyped Function

Use:
- Ctrl+L to set the function

**FunctionEdge**

Models the flow of control/objects through a function; used in SV-4.

Extensions:
- ControlFlow

Generalizations:
- SystemsElement

Specializations:
- SystemFunctionEdge

Constraints:
- Source must be a ServiceOperationAction
- Target must be a ServiceOperationAction

Use:
- Can realize a ResourceInteraction (Right-click | Advanced | Information Flows Realized)

**FunctionParameter**

Represents inputs and outputs of a Function; used in SV-4.

Extensions:
- ActivityParameter

Constraints:
- Type must be a ResourceInteractionItem (Energy, DataElement, CapabilityConfiguration, Organization, Post, ResourceArtifact or Software)
Use:
- Owned by a Function

**HighLevelOperationalConcept**

A generalized model for operations; used in OV-1.

Extensions:
- Class

Constraints:
- Owned attributes are ConceptRole

Use:
- Can have a set of described Mission

**HostedSoftware**

Asserts that software is hosted on a ResourceArtifact; used in SV-1.

Extensions:
- Part

Generalizations:
- ResourceRole

Constraints:
- Owning Class must be a ResourceArtifact
- Type must be a Software

Use: Can:
- Have a RequiresCompetence dependency to a Competence (from ResourceRole)
- Have a set of associations to 'used' Functions (from ResourceRole)

**HumanResource**

The role of a Post or Organization in a CapabilityConfiguration; used in SV-1.

Extensions:
- Part

Generalizations:
- ResourceRole

Constraints:
- Owning Class must be a CapabilityConfiguration
- Type must be an OrganizationalResource (Organization or Post)

Use: Can:
- Have a RequiresCompetence dependency to a Competence (from ResourceRole)
- Have a set of associations to 'used' Functions (from ResourceRole)
**ImplementsOperational**

Relationship between a system element and the operational element that it implements; used in SV-5.

Extensions:
- Dependency

Constraints:
- Client must be a SystemsElement (Function)
- Supplier must be an OperationalElement (OperationalActivity)

**IncrementMilestone**

An ActualProjectMilestone that indicates the point in time at which a project is predicted to deliver or has delivered a Capability; used in AcV-2, StV-3 and SV-8.

Extensions:
- Object

Generalizations:
- ActualProjectMilestone

Constraints:
- Classifier must be a ProjectMilestone (from ActualProjectMilestone)

Use:
- Can be the supplier or client of a MilestoneSequence dependency (from ActualProjectMilestone)
- Can have a set of associated Resource (from ActualProjectMilestone)
- Has a set of associations with CapabilityConfiguration

**InformationElement**

Information exchanged between nodes; used in OV-2, OV-3, OV-5, OV-6a, OV-6b and OV-7.

Extensions:
- Class

Generalizations:
- OperationalExchangeItem
- SubjectOfOperationalConstraint
- SubjectOfOperationalStateMachine
- OperationalElement

Use:
- Can:
  - Have a set of associations with (represented by) EntityItem Classes
  - Be conveyed on an InformationExchange - right-click > Advanced > Information Items Conveyed
  - Have an attached OperationalConstraint (from SubjectOfOperationalConstraint)
  - Own an OperationalStateMachine (from SubjectOfOperationalStateMachine)
**InformationExchange**

A relationship specifying the need to exchange information between nodes; used in OV-2, OV-3 and OV-6c.

Extensions:
- InformationFlow

Generalizations:
- OperationalExchange

Constraints:
- Conveys an InformationElement
- Source is a Node (from OperationalExchange)
- Target is a Node (from OperationalExchange)

**InternalDataModel**

DoDAF alias for PhysicalDataModel; used in SV-11.

Extensions:
- Package

Generalizations:
- PhysicalDataModel
- DataModel

Constraints:
- Owns EntityItem elements (from DataModel)

**KnownResource**

Asserts that a known resource plays a part in the architecture; used in OV-2.

Extensions:
- Part

Generalizations:
- NodeChild

Constraints:
- Type must be a Resource (Post, Organization, CapabilityConfiguration, SystemsNode, Software or ResourceArtifact)
- Class must be a NodeParent (Node or LogicalArchitecture) (from NodeChild)

**LightCondition**

A specification of environmental lighting conditions; used in StV-2.

Extensions:
- Class

Generalizations:
• EnvironmentalType  
  Use:  
  • Can be the type of an EnvironmentProperty (from EnvironmentalType)

Location

A general specification of the surroundings/scenario in which an operation might take place. Examples include 'desert', 'arctic', 'at sea'; used in OV-1 and OV-2.

Extensions:  
• Class  
Generalizations:  
• ReferredLocation  
• ConceptItem  
• EnvironmentalType  
Use: Can be:  
• Supplier to a CompatibleWith dependency from a Node (from ReferredLocation)  
• Type of a ConceptRole (from ConceptItem)  
• The type of an EnvironmentProperty (from EnvironmentalType)

LogicalArchitecture

A composite structure model whose parts are either NodeRoles, ProblemDomains, or KnownResources; used in OV-2.

Extensions:  
• Class  
Generalizations:  
• NodeParent  
Use:  
• Can own ProblemDomain properties

LogicalDataModel

A specification of business information requirements as a formal data structure; used in OV-7.

Extensions:  
• Package  
Generalizations:  
• DataModel  
Constraints:  
• Owns EntityItem elements (from DataModel)
Asserts that a StandardOperationalActivity is in some way part of a capability; used in StV-6.

Extensions:
- Dependency

Constraints:
- Client must be a StandardOperationalActivity
- Supplier must be a Capability

MaterielExchange

Materiel that is exchanged between Nodes; used in OV-2, OV-3 and OV-6c.

Extensions:
- InformationFlow

Generalizations:
- OperationalExchange

Constraints:
- Source is a Node (from OperationalExchange)
- Target is a Node (from OperationalExchange)

Use:
- Can convey a ResourceArtifact or Software

Measurement

A category of measures; used in AV-3, OV-2 and SV-7.

Extensions:
- Attribute

Specializations:
- PerformanceParameter

Use:
- Owned by a MeasurementSet Class

MeasurementSet

A set or collection of Measurements; used in AV-3, OV-3 and SV-7.

Extensions:
- Class

Constraints:
- Owned attributes must be Measurement

Use:
- Has a set of associations with (measuredElement) UPDMElement
- Is classifier of ActualMeasurementSet object
Metadata

Annotation that can be applied to any element in the architecture; used in AV-1.

Extensions:
- Note

Specializations:
- ArchitectureMetadata

MilestoneSequence

A relationship between two milestones; used in AcV-2 and SV-8.

Extensions:
- Dependency

Constraints:
- Client must be an ActualProjectMilestone
- Supplier must be an ActualProjectMilestone

Mission

A purpose to which a person, organization, or autonomous system is tasked; used in AV-1, OV-1, OV-6a and OV-6b.

Extensions:
- UseCase

Generalizations:
- SubjectOfOperationalConstraint
- SubjectOfOperationalStateMachine

Use:
- Fulfilled by an EnterprisePhase
- Can have an attached OperationalConstraint (from SubjectOfOperationalConstraint)
- Can own an OperationalStateMachine (from SubjectOfOperationalStateMachine)

MovementOfPeople

MODAF alias for OrganizationalExchange.

Extensions:
- InformationFlow

Generalizations:
- OrganizationalExchange

Use:
- Conveys an OrganizationalResource (Organization or Post)
**Needline**

Documents the requirement to exchange information between nodes; used in OV-2 and OV-3.

Extensions:
- Association
- Connector

Generalizations:
- OperationalElement

Constraints:
- End Types must be Node
- End Roles must be NodePort
- End Roles must be NodeChild (NodeRole, ProblemDomain, KnownResource)

Use:
- Realizes an OperationalExchange - create a Needline between the same two elements as an OperationalExchange, then right-click on the Needline and select 'Advanced > Information Flows Realized'

**NoLongerUsedMilestone**

 Asserts that an ActualOrganizationResource ceased to use - or is slated to cease using - a CapabilityConfiguration from a specific point in time; used in StV-5.

Extensions:
- Object

Generalizations:
- ActualProjectMilestone

Constraints:
- Classifier must be a ProjectMilestone (from ActualProjectMilestone)

Use:
- Has set of associations to 'noLongerUsedBy' ActualOrganizationalResource (ActualOrganization or ActualPost) objects
- Can have a set of associated Resource (from ActualProjectMilestone)
- Can be client/supplier of a MilestoneSequence (from ActualProjectMilestone)
- Has a set of associations with 'configuration' CapabilityConfiguration Classes

**Node**

Logical entity that performs operational activities; used in OV-1, OV-2, OV-3, OV-5, OV-6a, OV-6b and OV-6c.

Extensions:
- Class

Generalizations:
- Performer
- ConceptItem
- NodeParent
- SubjectOfOperationalConstraint
- SubjectOfOperationalStateMachine
- OperationalElement

Specializations:
- OperationalNode

Constraints:
- Owned ports must be NodePort, RequestPoint or ServicePoint

Use: Can:
- Have a Performs dependency to a PerformedActivity (Function or OperationalActivity) (from Performer)
- Be the Client of a CompatibleWith dependency to a ReferredLocation (Location or PhysicalLocation)
- Be the type of a ConceptRole (from ConceptItem)
- Own a RequestPoint port
- Own a ServicePoint port
- Be client of an ExhibitsCapability dependency to a Capability
- Own NodeChild (NodeRole, KnownResource, ProblemDomain) (from NodeParent)
- Be source and target of an OperationalExchange (ConfigurationExchange, EnergyExchange, InformationExchange, MaterielExchange or OrganizationalExchange) information flow
- Be the end type of a Needline association
- Have an attached OperationalConstraint (from SubjectOfOperationalConstraint)
- Own an OperationalStateMachine (from SubjectOfOperationalStateMachine)
- Be the type of a NodeRole
- Own ServiceOperations

**NodePort**

A property of a Node that specifies a distinct interaction point between the node and its environment or between the node and its internal parts.

Extensions:
- Port

Constraints:
- Type must be an OperationalExchangeItem (Post, Organization, ResourceArtifact or System)

Use:
- Owned by a Node
- Can be the ends of a Needline

**NodeRole**

Used to link a parent Node to its sub-nodes; used in OV-2, OV-3 and OV-6c.

Extensions:
- Part

Generalizations:
- NodeChild
Specializations:
- ProblemDomain

Constraints:
- Class must be a Node
- Type must be a Node

**OperationalActivity**

A logical process, specified independently of how the process is carried out; used in OV-2, OV-3, OV-4, OV-5, OV-6a, OV-6b and SV-5.

Extensions:
- Activity

Generalizations:
- PerformedActivity
- SubjectOfOperationalConstraint
- OperationalElement
- SubjectOfOperationalStateMachine

Specializations:
- StandardOperationalActivity

Constraints:
- Owned parameters must be OperationalParameter

Use: Can:
- Be Supplier of a Performs dependency (from PerformedActivity)
- Be Supplier of an OwnsProcess dependency
- Be the Activity/Behavior of an OperationalActivityAction
- Be the owner of an OperationalActivityEdge
- Have an attached OperationalConstraint (from SubjectOfOperationalConstraint)
- Be the Supplier of a SupportsOperationalActivity dependency
- Own an OperationalStateMachine (from SubjectOfOperationalStateMachine)

**OperationalActivityAction**

A call behavior action that invokes the activity to be performed; used in OV-5.

Extensions:
- CallBehaviorAction

Constraints:
- Activity/Behavior must be an OperationalActivity

Use:
- Can be the Source or Target of an OperationalActivityEdge

**OperationalActivityEdge**
Models the flow of control/objects through an OperationalActivity; used in OV-5.

Extensions:
- ControlFlow

Generalizations:
- OperationalElement

Constraints:
- Must be owned by an OperationalActivity
- Source must be an OperationalActivityAction
- Target must be an OperationalActivityAction

Use: Can:
- Have a set of OperationalExchange (ConfigurationExchange, EnergyExchange, InformationExchange, MaterielExchange or OrganizationalExchange) information flows that it realizes
- Carry a set of OperationalExchangeItem (Post, Organization, ResourceArtifact or System)

**OperationalConstraint**

A rule governing an operational behavior or property; used in OV-6a.

Extensions:
- Constraint

Specializations:
- OperationalRule

Constraints:
- Constrained element must be a SubjectOfOperationalConstraint (OperationalActivity, Node, InformationElement or Mission)

**OperationalMessage**

Message for use in an Operational Event Trace, which carries any of the subtypes of OperationalExchange; used in OV-6c.

Extensions:
- Message

Generalizations:
- OperationalElement

Use:
- Can have a set of OperationalExchange (ConfigurationExchange, EnergyExchange, InformationExchange, MaterielExchange or OrganizationalExchange) information flows that it realizes

**OperationalNode**

An element of the operational architecture that produces, consumes, or processes information.

Extensions:
- Class

Generalizations:
- Node

Constraints:
- Owned ports must be NodePort, RequestPoint or ServicePoint

Use: Can:
- Have a Performs dependency to a PerformedActivity (Function, OperationalActivity) (from Performer)
- Be the Client of a CompatibleWith dependency to a ReferredLocation (Location or PhysicalLocation)
- Be the type of a ConceptRole (from ConceptItem)
- Own a RequestPoint port
- Own a ServicePoint port
- Be client of an ExhibitsCapability dependency to a Capability
- Own NodeChild (NodeRole, KnownResource, ProblemDomain) (from NodeParent)
- Be source and target of an OperationalExchange (ConfigurationExchange, EnergyExchange, InformationExchange, MaterielExchange or OrganizationalExchange) information flow
- Be the end type of a Needline association
- Have an attached OperationalConstraint (from SubjectOfOperationalConstraint)
- Own an OperationalStateMachine (from SubjectOfOperationalStateMachine)
- Be type of a NodeRole
- Own ServiceOperations

**OperationalParameter**

Represents inputs and outputs of an operational activity; used in OV-5.

Extensions:
- ActivityParameter

Constraints:
- Type must be an OperationalExchangeItem (Post, Organization, ResourceArtifact or System)

Use:
- Can be owned by an OperationalActivity

**OperationalRule**

A DoDAF alias for OperationalConstraint.

Extensions:
- Constraint

Generalizations:
- OperationalConstraint

Constraints:
- Constrained element must be a SubjectOfOperationalConstraint (OperationalActivity, Node, InformationElement or Mission) (from OperationalConstraint)
OperationalStateMachine

A StateMachine describing an operational behavior or property; used in OV-6b.

Extensions:
- StateMachine

Constraints:
- Owner is SubjectOfOperationalStateMachine (Mission, InformationElement or Node)

Organization

A group of persons, associated for a particular purpose; used in OV-4, SV-1, SV-3, SV-9, SV-10a and SV-12.

Extensions:
- Class

Generalizations:
- OrganizationalResource
- Resource, Performer
- SubjectOfForecast
- SubjectOfResourceConstraint

Use: Can:
- Be classifier to an ActualOrganization
- Be source or target of a Commands information flow (from OrganizationalResource)
- Be the owning Class of a PostRole
- Be the Class or type of a SubOrganization
- Be the Class of an Equipment (from OrganizationalResource)
- Be conveyed by an OrganizationalExchange (from OrganizationalResource)
- Be the type of a HumanResource (from OrganizationalResource)
- Be the source of a Controls information flow (from OrganizationalResource)
- Have a set of associated milestones, stereotyped ActualProjectMilestone (from Resource)
- Be the client of a RealizesCapability realization to a Capability (from Resource)
- Be the client of a ProvidesCompetence dependency to a Competence (from Resource)
- Have an attached ResourceConstraint (from Resource, SubjectOfResourceConstraint)
- Be supplier or client of a Forecast dependency (both must have same stereotype) (from SubjectOfForecast)
- Own a ServicePoint (from Resource)
- Own a RequestPoint (from Resource)
- Own a ResourcePort (from Resource)
- Be source and target of a ResourceInteraction (from Resource)
- Own a ServiceOperation (from Resource)
- Be type of a KnownResource (from Resource)
- Be type of a ResourceRole (from Resource)
- Have a Performs dependency to a PerformedActivity (Function or OperationalActivity) (from Performer)
**OrganizationalExchange**

A relationship specifying flow of people across organizations; used in OV-2, OV-3 and OV-6c.

Extensions:
- InformationFlow

Generalizations:
- OperationalExchange

Specializations:
- MovementOfPeople

Constraints:
- Conveyed element must be an OrganizationalResource (Organization or Post)
- Source is a Node (from OperationalExchange)
- Target is a Node (from OperationalExchange)

**OutOfServiceMilestone**

A project milestone that indicates a project's deliverable is to go out of service; used in AcV-2, StV-3 and SV-8.

Extensions:
- Object

Generalizations:
- ActualProjectMilestone

Constraints:
- Classifier must be a ProjectMilestone

Use:
- Has a set of association ('configuration') with CapabilityConfiguration
- Can have a set of associated Resource (from ActualProjectMilestone)
- Can be client/supplier of a MilestoneSequence (from ActualProjectMilestone)

**OwnsProcess**

A relationship that asserts that an ActualOrganizationalResource has responsibility for an OperationalActivity; used in OV-4.

Extensions:
- Dependency

Constraints:
- Client must be an ActualOrganizationalResource (ActualPost or ActualOrganization)
- Supplier must be an OperationalActivity

**Part**

Use of a ResourceArtifact as a part of another ResourceArtifact; used in SV-1.
Extensions:
- Part

Generalizations:
- ResourceRole

Specializations:
- SubSystemPart

Constraints:
- Class must be a ResourceArtifact
- Type must be a ResourceArtifact

Use: Can have:
- A RequiresCompetence dependency to a Competence (from ResourceRole)
- A set of associations to 'used' Functions (from ResourceRole)

**PerformanceParameter**

A category of quality measures that address how well a Performer meets Capability needs.

Extensions:
- Attribute

Generalizations:
- Measurement

Use:
- Owned by a MeasurementSet class

**Performs**

Links a Performer to the behavior that it can perform; used in OV-2, OV-3, OV-4, OV-5, SV-1 and SV-4.

Extensions:
- Dependency

Constraints:
- Client must be a Performer (Node, ExternalNode, OperationalNode, Post, Organization, CapabilityConfiguration, SystemsNode, Software or ResourceArtifact)
- Supplier must be a PerformedActivity (OperationalActivity or Function)

**Person**

A type of human being; used in OV-4.

Extensions:
- Class

Use:
- Can be Classifier of an ActualPerson
PhysicalDataModel

An implementable specification of a data structure; used in SV-11.

Extensions:
- Package

Generalizations:
- DataModel

Specializations:
- InternalDataModel

Constraints:
- Owns EntityItem elements (from DataModel)

PhysicalLocation

Anywhere that can be specified; used in OV-1 and OV-2.

Extensions:
- Class

Generalizations:
- ReferredLocation
- ConceptItem
- EnvironmentalType

Use: Can be:
- Supplier to a CompatibleWith dependency from a Node (from ReferredLocation)
- Type of a ConceptRole (from ConceptItem)
- The type of an EnvironmentProperty (from EnvironmentalType)

Platform

Use of an artifact as a platform in a particular ResourceConfiguration; used in SV-1.

Extensions:
- Part

Generalizations:
- ResourceComponent
- ResourceRole

Constraints:
- Class must be a CapabilityConfiguration
- Type must be a ResourceArtifact

Use:
- Can have a RequiresCompetence dependency to a Competence (from ResourceRole)
- Can have a set of associations to 'used' Functions (from ResourceRole)
Post

A type of point of contact or responsible person; used in OV-4, SV-1, SV-3, SV-9, SV-10a and SV-12.

Extensions:
- Class

Generalizations:
- OrganizationalResource
- Resource
- Performer
- SubjectOfForecast
- SubjectOfResourceConstraint

Use: Can:
- Be Classifier of an ActualPost
- Be the Type of a PostRole
- Be source or target of a Commands information flow (from OrganizationalResource)
- Be the Class of an Equipment (from OrganizationalResource)
- Be conveyed by an OrganizationalExchange (from OrganizationalResource)
- Be the type of a HumanResource (from OrganizationalResource)
- Be the source of a Controls information flow (from OrganizationalResource)
- Have a set of associated milestones, stereotyped ActualProjectMilestone (from Resource)
- Be client of a RealizesCapability realization to a Capability (from Resource)
- Be client of a ProvidesCompetence dependency to a Competence (from Resource)
- Have an attached ResourceConstraint (from Resource, SubjectOfResourceConstraint)
- Be supplier or client of a Forecast dependency (both must have same stereotype) (from SubjectOfForecast)
- Own a ServicePoint (from Resource)
- Own a RequestPoint (from Resource)
- Own a ResourcePort (from Resource)
- Be source and target of a ResourceInteraction (from Resource)
- Own a ServiceOperation (from Resource)
- Be type of a KnownResource (from Resource)
- Be type of a ResourceRole (from Resource)
- Have a Performs dependency to a PerformedActivity (Function, OperationalActivity) (from Performer)

PostRole

Asserts that a post exists in an organization; used in OV-4 and SV-1.

Extensions:
- Part

Generalizations:
- OrganizationRole
- ResourceRole
Constraints:
- Class must be an Organization
- Type must be a Post

Use: Can have a:
- RequiresCompetence dependency to a Competence (from ResourceRole)
- Set of associations to 'used' Functions (from ResourceRole)

ProblemDomain

The boundary containing those Nodes that can be realized by functional resources; used in OV-2.

Extensions:
- Part

Generalizations:
- NodeRole
- NodeChild

Constraints:
- Class must be a LogicalArchitecture
- Type must be a Node (from NodeRole)

Project

Used to define a category of project; used in AcV-1.

Extensions:
- Class

Use: Can:
- Be classifier of an ActualProject
- Have an association to a ProjectMilestone Class

ProjectMilestone

A type of project milestone; used in AcV-2.

Extensions:
- Class

Constraints:
- Owned attributes must be ProjectTheme

Use: Can:
- Be classifier of an ActualProjectMilestone
- Have an association from a Project Class

ProjectSequence
Asserts that one ActualProject follows on from another; used in AcV-2.
Extensions:
- Dependency
Constraints:
- Client must be an ActualProject
- Supplier must be an ActualProject

ProjectTheme

An aspect by which the progress of various projects can be measured; used in AcV-2.
Extensions:
- Attribute
Constraints:
- Type must be a ProjectThemeStatus
Use:
- Owned by ProjectMilestone

ProjectThemeStatus

Specifies a status for a ProjectTheme.
Extensions:
- Class
Use:
- The type of a ProjectTheme

Protocol

A standard for communication; used in SV-2, TV-1 and TV-2.
Extensions:
- Class
Generalizations:
- Standard
- SubjectOfForecast
Use: Can:
- Have a set of associations with ('ratifiedBy') ActualOrganization objects (from Standard)
- Have ProtocolLayers
- Be the type of ProtocolLayers
- Be the client and supplier of a Forecast dependency
**ProtocolLayer**

Asserts that a protocol uses another protocol; used in TV-1 and TV-2.

Extensions:
- Attribute

Constraints:
- Owning Class must be a Protocol
- Type must be a Protocol

**ProvidesCompetence**

Asserts that a resource provides a competence; used in OV-4.

Extensions:
- Dependency

Constraints:
- Client must be a Resource (Post, Organization, CapabilityConfiguration, SystemsNode, Software or ResourceArtifact)
- Supplier must be a Competence

**RealizesCapability**

Asserts that a resource provides a capability; used in SOV-3, StV-3, StV-5 and SV-1.

Extensions:
- Realization

Constraints:
- Client must be a Resource or a ServiceInterface
- Supplier must be a Capability

**RequestPoint**

The mechanism by which a Service communicates; used in OV-2 and SV-1.

Extensions:
- Port

Constraints:
- Type must be a ServiceInterface

Use:
- Can be owned by a Node or a Resource

**RequiresCompetence**

Asserts that a role requires a competence; used in SV-1.
Extensions:
- Dependency

Constraints:
- Client must be a ResourceRole
- Supplier must be a Competence

**ResourceArtifact**

A type of man-made object; used in OV-2, OV-3, OV-5, SV-1, SV-3, SV-9, SV-10a and SV-12.

Extensions:
- Class

Generalizations:
- OperationalExchangeItem
- ManufacturedResourceType
- Resource
- SubjectOfForecast
- ResourceInteractionItem
- Performer
- SubjectOfResourceConstraint

Specializations:
- System

Use: Can:
- Be conveyed by a MaterielExchange
- Be the type of an OperationalParameter (from OperationalExchangeItem)
- Own HostedSoftware
- Be the Class and type of a Part
- Be the type of a ResourceComponent
- Be the type of an Equipment
- Be the target of a Controls flow (from ManufacturedResourceType)
- Have a set of associated milestones, stereotyped ActualProjectMilestone (from Resource)
- Be client of a RealizesCapability realization to a Capability (from Resource)
- Be client of a ProvidesCompetence dependency to a Competence (from Resource)
- Have an attached ResourceConstraint (from Resource, SubjectOfResourceConstraint)
- Be supplier or client of a Forecast dependency (both must have same stereotype) (from SubjectOfForecast)
- Own a ServicePoint (from Resource)
- Own a RequestPoint (from Resource)
- Own a ResourcePort (from Resource)
- Be source and target of a ResourceInteraction (from Resource)
- Own a ServiceOperation (from Resource)
- Be type of a KnownResource (from Resource)
- Be type of a ResourceRole (from Resource)
- Have a Performs dependency to a PerformedActivity (Function or OperationalActivity) (from Performer)
**ResourceComponent**

A well-defined resource that is used by a CapabilityConfiguration to accomplish a capability; used in SV-1.

Extensions:
- Part

Generalizations:
- ResourceRole

Specializations:
- Platform

Constraints:
- Type must be a ResourceArtifact
- Owning Class must be a CapabilityConfiguration

Use: Can have a:
- RequiresCompetence dependency to a Competence (from ResourceRole)
- Set of associations to 'used' Functions (from ResourceRole)

**ResourceConnector**

A physical connection between two resources that implements protocols through which the source resource can transmit items to the destination resource; used in SV-2.

Extensions:
- Connector

Generalizations:
- ProtocolImplementation

Constraints:
- End roles must be ResourcePort

Use: Can:
- Have a set of ResourceInterface that it realizes
- Realize a ResourceInteraction

**ResourceConstraint**

Specifies the set of rules that govern the structural or functional aspects of the system; used in SV-10a.

Extensions:
- Constraint

Constraints:
- Constrained element must be a SubjectOfResourceConstraint (DataElement, Function, SystemFunction, CapabilityConfiguration, SystemsNode, Software, ResourceArtifact, System, Post or Organization)
Represents data that is exchanged between resources; used in OV-4, SOV-4c, SV-1, SV-2, SV-3, SV-4, SV-6 and SV-10c.

Extensions:
- InformationFlow

Generalizations:
- SystemsElement
- ProtocolImplementation

Specializations:
- Controls
- Commands
-DataExchange

Constraints:
- Realizing connector is a ResourceInterface
- Realizing activity edge is a FunctionEdge
- Conveyed elements must be ResourceInteractionItem (DataElement, Energy, Post, Organization, CapabilityConfiguration, SystemsNode, Software, ResourceArtifact or System)
- Source must be a Resource (Post, Organization, CapabilityConfiguration, SystemsNode, Software, ResourceArtifact or System)
- Target must be a Resource (Post, Organization, CapabilityConfiguration, SystemsNode, Software, ResourceArtifact or System)

Use:
- Can realize an OperationalExchange (OrganizationalExchange, InformationExchange, EnergyExchange or MaterielExchange)
- Can realize an ActualOrganizationRelationship
- Has an association to ('implements') a Protocol (from ProtocolImplementation)

**ResourceInterface**

A contractual agreement between two resources that implement protocols; used in OV-4, SV-1, SV-2, SV-3 and SV-6.

Extensions:
- Association
- Connector

Generalizations:
- SystemsElement

Specializations:
- SystemConnector

Constraints:
- End roles must be ResourceRole
- End types must be Resource

Use:
- Can realize a ResourceInteraction
ResourceMessage

Message for use in a Resource event trace, implements a ResourceInteraction; used in SV-10c.
Extensions:
- Message
Generalizations:
- SystemsElement
Use:
- Can have a set of ResourceInteraction that it carries

ResourcePort

An interaction point for a resource through which it can interact with the outside environment; used in SV-2.
Extensions:
- Port
Generalizations:
- ProtocolImplementation
Constraints:
- Type must be a ResourceInteractionItem (Energy, Post, Organization, CapabilityConfiguration, Software, ResourceArtifact or DataElement)
Use:
- Can be owned by a Resource
- Has an association to a Protocol Class that it 'implements' (from ProtocolImplementation)
- Can be the end role of a ResourceConnector

ResourceStateMachine

UPDM artifact that extends a UML StateMachine applied to Resources; used in SV-10b.
Extensions:
- StateMachine
Generalizations:
- SystemsElement
Constraints:
- Owner must be SubjectOfResourceStateMachine (Post, Organization, CapabilityConfiguration, SystemsNode, Software, ResourceArtifact, System or DataElement)

SameAs

Asserts that two elements refer to the same real-world thing; used in AV-2.
Extensions:
- Dependency
Constraints:
- Client must be a UPDMElement
- Supplier must be an ExternalIndividual or ExternalType

**ServiceAttribute**

A property of a ServiceInterface that allows performance, reliability and cost values to be captured; used in SOV-1.

Extensions:
- Attribute

Use:
- Owned by a ServiceInterface

**ServiceFunction**

Describes the abstract behavior of ServiceOperations, regardless of the actual implementation; used in SOV-5.

Extensions:
- Activity

Use: Can:
- Be the behavior of a ServiceFunctionAction
- Be the activity of a ServiceOperationAction
- Own ServicePoint ports

**ServiceFunctionAction**

A call behavior action that invokes the ServiceFunction to be performed; used in SOV-5.

Extensions:
- CallBehaviorAction

Constraints:
- Behavior must be a ServiceFunction

**ServiceInteraction**

Interaction for a service interface; used in SOV-4c.

Extensions:
- Interaction

**ServiceInterface**

A contractual agreement between two resources that implement protocols through which the source service interacts with the destination resource; used in SOV-1, SOV-2, SOV-3, SOV-4a, SOV-4b, SOV-4c and SOV-5.

Extensions:
• Class
Constraints:
• Owned attributes must be ServiceAttribute
• Owned operations must be ServiceOperation
Use: Can:
• Be client of a SupportsOperationalActivity dependency to an OperationalActivity
• Be client of a RealizesCapability realization to a Capability
• Own ServicePolicy
• Have one association to a ServiceStateMachine
• Have one association to a ServiceInteraction
• Be type of a RequestPoint or ServicePoint port
• Be dependent on another ServiceInterface
• Be client of an Expose dependency to a Capability

ServiceMessage

Message for use in a service interaction specification, implements a resource interaction; used in SOV-4c.
Extensions:
• Message
Use:
• Can carry a set of ResourceInteractions

ServiceOperation

Provides the access point for invoking the behavior of a provided service; used in SOV-2 and SOV-5.
Extensions:
• Operation
Constraints:
• Owner must be a Resource (Post, Organization, CapabilityConfiguration, SystemsNode, Software, ResourceArtifact or System)
• Owner must be a Node
Use: Can:
• Have an association to a (concreteBehavior) Function
• Be owned by a ServiceInterface
• Be the operation of a ServiceOperationAction
• Have an association to an (abstractBehavior) ServiceFunction

ServiceOperationAction

A call action that represents a Resource or ServiceFunction invoking a ServiceOperation; used in SOV-5.
Extensions:
- CallOperationAction

Constraints:
- Activity must be a ServiceFunction
- Activity must be a Function
- Operation must be a ServiceOperation

Use:
- Can be the Source and Target of a FunctionEdge control flow

**ServicePoint**

The mechanism by which a service communicates; used in OV-2, SV-1 and SV-12.

Extensions:
- Port

Constraints:
- Type must be a ServiceInterface
- Owned behavior is a ServiceFunction

Use:
- Can be owned by a Node or a Resource (Post, Organization, CapabilityConfiguration, SystemsNode, Software, ResourceArtifact or System)

**ServicePolicy**

A constraint governing the consumers and providers of services; used in SOV-4a.

Extensions:
- Constraint

Use:
- Rule can be owned by a ServiceInterface

**ServiceStateMachine**

UPDM artifact that extends UML StateMachine; used in SOV-4b.

Extensions:
- StateMachine

**Software**

Software needed for the functioning of the system; used in OV-2, OV-3, SV-1, SV-3, SV-9, SV-10a and SV-12.

Extensions:
- Class

Generalizations:
- ManufacturedResourceType
• Resource
• SubjectOfForecast
• ResourceInteractionItem
• Performer
• SubjectOfResourceConstraint

Use: Can:
• Be conveyed on a MaterielExchange information flow
• Be type of HostedSoftware
• Be the target of a Controls flow (from ManufacturedResourceType)
• Have a set of associated milestones, stereotyped ActualProjectMilestone (from Resource)
• Be client of a RealizesCapability realization to a Capability (from Resource)
• Be client of a ProvidesCompetence dependency to a Competence (from Resource)
• Have an attached ResourceConstraint (from Resource, SubjectOfResourceConstraint)
• Be supplier or client of a Forecast dependency (both must have same stereotype) (from SubjectOfForecast)
• Own a ServicePoint (from Resource)
• Own a RequestPoint (from Resource)
• Own a ResourcePort (from Resource)
• Be source and target of a ResourceInteraction (from Resource)
• Own a ServiceOperation (from Resource)
• Be a type of a KnownResource (from Resource)
• Be a type of a ResourceRole (from Resource)
• Have a Performs dependency to a PerformedActivity (Function or OperationalActivity) (from Performer)

**Standard**

A ratified set of rules that are used to guide and/or constrain any UPDM element; used in SV-9, TV-1 and TV-2.

Extensions:
• Class

Generalizations:
• SubjectOfForecast

Specializations:
• Protocol

Use:
• Any UPDMElement can have a 'conformsTo' association to a Standard
• Can have an association (ratifiedBy) with an ActualOrganization
• Can be supplier or client of a Forecast (both must be same stereotype) (from SubjectOfForecast)

**StandardConfiguration**

A comment, attached to a CapabilityConfiguration, indicating that the annotated CapabilityConfiguration is a standard Pattern for re-use in the architecture; used in TV1 and TV-2.

Extensions:
- **Note**
  Constraints:
  - The annotated element must be a CapabilityConfiguration

**StandardOperationalActivity**

An OperationalActivity that is a standard procedure and that is doctrinal; used in OV-5 and StV-6.

Extensions:
- Activity

Generalizations:
- OperationalActivity
- PerformedActivity
- SubjectOfOperationalConstraint
- OperationalElement
- SubjectOfOperationalStateMachine

Constraints:
- Owned parameters must be OperationalParameter (from OperationalActivity)

Use: Can:
- Be Client of a MapsToCapability dependency to a Capability Class
- Be Supplier of a Performs dependency (from PerformedActivity)
- Be Supplier of an OwnsProcess dependency (from OperationalActivity)
- Be the Activity/Behavior of an OperationalActivityAction (from OperationalActivity)
- Be the owner of an OperationalActivityEdge (from OperationalActivity)
- Have an attached OperationalConstraint (from SubjectOfOperationalConstraint)
- Be the Supplier of a SupportsOperationalActivity dependency (from OperationalActivity)
- Own an OperationalStateMachine (from SubjectOfOperationalStateMachine)

**StereotypeExtension**

Defines an additional stereotype used in the architecture that is not defined in this metamodel; used in AV-2.

Extensions:
- Note

Constraints:
- Annotated element must be a UPDMElement

Use:
- Can have a set of associations (ontologyReference) to ExternalType

**StructuralPart**

Describes a structural part of an EnterprisePhase; used in AV-1.

Extensions:
- Part
  Constraints:
  - Type must be an EnterprisePhase
  - Class must be an EnterprisePhase

SubOrganization

Asserts that one type of organization is typically the parent of another; used in OV-4 and SV-1.

Extensions:
- Part
Generalizations:
  - OrganizationRole
  - ResourceRole
Constraints:
  - Type must be an Organization
  - Class must be an Organization
Use: Can:
  - Have a RequiresCompetence dependency to a Competence (from ResourceRole)
  - Have a set of associations to 'used' Functions (from ResourceRole)

SubSystemPart

Indicates that a subsystem is part of another system; used in SV-1.

Extensions:
- Part
Generalizations:
  - Part
  - ResourceRole
Constraints:
  - Class must be a ResourceArtifact (from Part)
  - Type must be a ResourceArtifact (from Part)
Use: Can:
  - Have a RequiresCompetence dependency to a Competence (from ResourceRole)
  - Have a set of associations to 'used' Functions (from ResourceRole)

SupportsOperationalActivity

An assertion that a Service in some way contributes or assists in the execution of an OperationalActivity.

Extensions:
- Dependency
Constraints:
• Client must be a ServiceInterface
• Supplier must be an OperationalActivity

System

Any organized assembly of resources and procedures united and regulated by interaction of interdependence to accomplish a set of specific functions.

Extensions:
• Class

Generalizations:
• ResourceArtifact
• OperationalExchangeItem
• ManufacturedResourceType
• Resource
• SubjectOfForecast
• ResourceInteractionItem
• Performer
• SubjectOfResourceConstraint

Use: Can:
• Be conveyed by a MaterielExchange (from ResourceArtifact)
• Be the type of an OperationalParameter (from OperationalExchangeItem)
• Own HostedSoftware (from ResourceArtifact)
• Be the Class and type of a Part (from ResourceArtifact)
• Be the type of a ResourceComponent (from ResourceArtifact)
• Be the type of an Equipment (from ResourceArtifact)
• Be the target of a Controls flow (from ManufacturedResourceType)
• Have a set of associated milestones, stereotyped ActualProjectMilestone (from Resource)
• Be client of a RealizesCapability realization to a Capability (from Resource)
• Be client of a ProvidesCompetence dependency to a Competence (from Resource)
• Have an attached ResourceConstraint (from Resource, SubjectOfResourceConstraint)
• Be supplier or client of a Forecast dependency (both must have same stereotype) (from SubjectOfForecast)
• Own a ServicePoint (from Resource)
• Own a RequestPoint (from Resource)
• Own a ResourcePort (from Resource)
• Be source and target of a ResourceInteraction (from Resource)
• Own a ServiceOperation (from Resource)
• Be type of a KnownResource (from Resource)
• Be type of a ResourceRole (from Resource)
• Have a Performs dependency to a PerformedActivity (Function or OperationalActivity) (from Performer)

SystemConnector
A link between two systems.

Extensions:
- Association
- Connector

Generalizations:
- ResourceInterface
- SystemsElement

Specializations:
- SystemConnector

Constraints:
- End roles must be ResourceRole (from ResourceInterface)
- End types must be Resource (from ResourceInterface)

Use:
- Can realize a ResourceInteraction (from ResourceInterface)

**SystemFunction**

A DoDAF alias for Function.

Extensions:
- Activity

Generalizations:
- Function
- PerformedActivity
- SystemsElement
- SubjectOfResourceConstraint

Constraints:
- Owned parameters are FunctionParameter (from Function)

Use: Can:
- Be Supplier of a Performs dependency (from PerformedActivity)
- Own ServiceOperationAction, FunctionAction or FunctionEdge (from Function)
- Be Client of an ImplementsOperational dependency to an OperationalActivity (from SystemsElement)
- Have an attached ResourceConstraint (from SubjectOfResourceConstraint)

**SystemFunctionAction**

A DoDAF alias for FunctionAction.

Extensions:
- CallBehaviorAction

Generalizations:
- FunctionAction

Constraints:
• Activity is stereotyped Function (from FunctionAction)
Use:
• Press Ctrl+L to set the function (from FunctionAction)

**SystemFunctionEdge**

An alias for FunctionEdge.

Extensions:
• A DoDAF ControlFlow

Generalizations:
• FunctionEdge
• SystemsElement

Constraints:
• Source must be a ServiceOperationAction (from FunctionEdge)
• Target must be a ServiceOperationAction (from FunctionEdge)

Use:
• Can realize a ResourceInteraction (right-click, Advanced > Information Flows Realized) (from FunctionEdge)

**SystemsNode**

A DoDAF alias for CapabilityConfiguration.

Extensions:
• Class

Generalizations:
• CapabilityConfiguration
• Resource, ConceptItem
• Performer
• ResourceInteractionItem
• SubjectOfResourceConstraint
• SubjectOfForecast
• SystemsElement
• SubjectOfResourceStateMachine
• ResourceInteractionItem

Use:
• Can:
  • Have a set of associated deployed milestones, stereotyped DeployedMilestone (from CapabilityConfiguration)
  • Have an optional associated no longer used milestone, stereotyped NoLongerUsedMilestone (from CapabilityConfiguration)
  • Have a set of associated increment milestones, stereotyped IncrementMilestone (from CapabilityConfiguration)
  • Have an optional associated out of service milestone, stereotyped OutOfServiceMilestone (from CapabilityConfiguration)
  • Be annotated by a StandardConfiguration note (from CapabilityConfiguration)
  • Be the type of a ConceptRole (from ConceptItem)
• Have a set of associated milestones, stereotyped ActualProjectMilestone (from Resource)
• Be client of a RealizesCapability realization to a Capability (from Resource)
• Be client of a ProvidesCompetence dependency to a Competence (from Resource)
• Have an attached ResourceConstraint (from Resource, SubjectOfResourceConstraint)
• Be supplier or client of a Forecast dependency (both must have same stereotype) (from SubjectOfForecast)
• Own a ServicePoint (from Resource)
• Own a RequestPoint (from Resource)
• Own a ResourcePort (from Resource)
• Be source and target of a ResourceInteraction (from Resource)
• Own a ServiceOperation (from Resource)
• Be the type of a KnownResource (from Resource)
• Be the type of a ResourceRole (from Resource)
• Have a Performs dependency to a PerformedActivity (Function, OperationalActivity) (from Performer)

**TechnologyForecast**

A statement about the future state of one or more types of standard.

Extensions:
• Forecast
• Dependency

Constraints:
• Client and Supplier are both SubjectOfForecast (Standard, Competence, Capability, CapabilityConfiguration, Organization, Post, ResourceArtifact or Software) (from Forecast)
• Client and Supplier must be the same specialization of SubjectOfForecast (from Forecast)

**TemporalPart**

EnterprisePhase elements that have a time-based nature; used in AV-1.

Extensions:
• Part

Constraints:
• Type must be an EnterprisePhase
• Class must be an EnterprisePhase

**UsedConfiguration**

The use of a CapabilityConfiguration in another CapabilityConfiguration; used in SV-1.

Extensions:
• Part

Generalizations:
• ResourceRole
Constraints:
- Type must be a CapabilityConfiguration
- Class must be a CapabilityConfiguration

Use: Can:
- Have a RequiresCompetence Dependency to a Competence (from ResourceRole)
- Have a set of Associations (usedFunctions) to Function (from ResourceRole)

**VisionStatement**

A high-level textual description of an EnterpriseVision.

Extensions:
- Note

**WholeLifeEnterprise**

A purposeful endeavor of any size involving people, organizations and supporting systems; used in AV-1 and StV-1.

Extensions:
- Class

Generalizations:
- EnterprisePhase

Use: Can:
- Have a set of Associations (statementTasks) to EnduringTask Class (from EnterprisePhase)
- Have a set of Associations (exhibits) to Capability Class (from EnterprisePhase)
- Have a set of Associations (inhabits) to Environment Class (from EnterprisePhase)
- Have a set of Associations (goals) with EnterpriseGoal Class (from EnterprisePhase)
- Have a set of Associations (visions) with EnterpriseVision Class (from EnterprisePhase)
- Be the type of a StructuralPart or TemporalPart (from EnterprisePhase)
- Fulfill a Mission Use Case (from EnterprisePhase)
- Be Supplier of a DefinesArchitecture Realization (from EnterprisePhase)
## Abstract Stereotypes

### Stereotype Specializations

<table>
<thead>
<tr>
<th>Stereotype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActualOrganizationalResource</td>
<td>An actual organization or post. Specializations:</td>
</tr>
<tr>
<td></td>
<td>• ActualOrganization</td>
</tr>
<tr>
<td></td>
<td>• ActualPost</td>
</tr>
<tr>
<td>ConceptItem</td>
<td>An item that might feature in a high level operational concept. Specializations:</td>
</tr>
<tr>
<td></td>
<td>• CapabilityConfiguration</td>
</tr>
<tr>
<td></td>
<td>• Node</td>
</tr>
<tr>
<td></td>
<td>• ReferredLocation</td>
</tr>
<tr>
<td></td>
<td>• Resource</td>
</tr>
<tr>
<td>DataModel</td>
<td>A structured specification of data, showing classifications of data elements and the relationships between them. Specializations:</td>
</tr>
<tr>
<td></td>
<td>• LogicalDataModel</td>
</tr>
<tr>
<td></td>
<td>• PhysicalDataModel</td>
</tr>
<tr>
<td>EnvironmentalType</td>
<td>A type of environment. Specializations:</td>
</tr>
<tr>
<td></td>
<td>• LightCondition</td>
</tr>
<tr>
<td></td>
<td>• Location</td>
</tr>
<tr>
<td></td>
<td>• PhysicalLocation</td>
</tr>
<tr>
<td></td>
<td>• Climate</td>
</tr>
<tr>
<td>ManufacturedResourceType</td>
<td>A resource artifact or software. Generalizations:</td>
</tr>
<tr>
<td></td>
<td>• Resource</td>
</tr>
<tr>
<td></td>
<td>Specializations:</td>
</tr>
<tr>
<td></td>
<td>• ResourceArtifact</td>
</tr>
<tr>
<td></td>
<td>• Software</td>
</tr>
<tr>
<td>NodeChild</td>
<td>An abstract element used for supporting the composite structuring of operational elements such as Nodes and LogicalArchitectures. Specializations:</td>
</tr>
<tr>
<td></td>
<td>• NodeRole</td>
</tr>
<tr>
<td></td>
<td>• ProblemDomain</td>
</tr>
<tr>
<td></td>
<td>• KnownResource</td>
</tr>
</tbody>
</table>
| NodeParent | Represents the owners/context of composite structure at the operational level. Specializations:  
| Node  
| ExternalNode  
| OperationalNode  
| LogicalArchitecture |
| OperationalElement | Elements relating to operational models. Specializations:  
| OperationalActivity  
| StandardOperationalActivity  
| OperationalMessage  
| Node  
| ExternalNode  
| OperationalNode  
| Needline  
| OperationalExchange  
| InformationElement  
| OperationalActivityEdge |
| OperationalExchange | Describes the characteristics of an exchanged item, such as the content, format (voice, imagery, text and message format), throughput requirements, security or classification level, timeliness requirement, and the degree of interoperability. Generalizations:  
| OperationalElement  
| Specializations:  
| ConfigurationExchange  
| EnergyExchange  
| InformationExchange  
| MaterielExchange  
| OrganizationalExchange |
| OperationalExchangeItem | An item that participates in an operational exchange. Specializations:  
| Post  
| Organization  
| ResourceArtifact  
| System |
| OrganizationalResource | Either an organization or a post. Generalizations:  
| Resource  
| OperationalExchangeItem  
| Specializations:  
| Post  
| Organization |
| OrganizationRole | Represents properties in an organization that are typed by another organization or a post.  
|                 | Generalizations:  
|                 | • ResourceRole  
|                 | Specializations:  
|                 | • SubOrganization  
|                 | • PostRole  
| PerformedActivity | A behavior that can be performed by a Performer.  
|                 | Specializations:  
|                 | • OperationalActivity  
|                 | • Function  
| Performer | A structural element that can perform behaviors (such as PerformedActivity)  
|             | Specializations:  
|             | • Node  
|             | • Resource  
| ProtocolImplementation | An element that implements a specific protocol.  
|                 | Specializations:  
|                 | • ResourcePort  
|                 | • ResourceInteraction  
|                 | • Controls  
|                 | • Commands  
|                 | • DataExchange  
|                 | • ResourceConnector  
| ReferredLocation | Either an actual location or a type of location (that is, environment) at/in which operations can be conducted.  
|                 | Generalizations:  
|                 | • ConceptItem  
|                 | • EnvironmentalType  
|                 | Specializations:  
|                 | • Location  
|                 | • PhysicalLocation  
| Resource | A physical asset, organizational resource or functional resource that can contribute towards fulfilling a capability.  
|             | Generalizations:  
|             | • SystemsElement  
|             | • SubjectOfResourceStateMachine  
|             | • ResourceInteractionItem  
|             | • Performer  
|             | • SubjectOfResourceConstraint  
|             | • ConceptItem  
|             | • SubjectOfForecast  

(c) Sparx Systems 2020
Page 155 of 328
Created with Enterprise Architect
<table>
<thead>
<tr>
<th>Concept</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ResourceInteractionItem</strong></td>
<td>Represents the items exchanged between resources through a resource interaction.</td>
</tr>
<tr>
<td><strong>ResourceRole</strong></td>
<td>Defines the usage of any resource in the system.</td>
</tr>
<tr>
<td><strong>SubjectOfForecast</strong></td>
<td>Any element that can be subject to a forecast.</td>
</tr>
<tr>
<td><strong>SubjectOfOperationalConstraint</strong></td>
<td>An element of the architecture that can be subject to an OperationalConstraint or OperationalStateDescription.</td>
</tr>
</tbody>
</table>
| **SubjectOfOperationalStateMachine** | The element being described by the StateMachine. Specializations:  
- OperationalActivity  
- InformationElement  
- Node  
- Mission |
| **SubjectOfResourceConstraint** | Anything that can be constrained by a ResourceConstraint. Specializations:  
- Post  
- Organization  
- CapabilityConfiguration  
- SystemsNode  
- Software  
- ResourceArtifact  
- System  
- DataElement  
- Function |
| **SubjectOfResourceStateMachine** | The element being described by the StateMachine. Specializations:  
- Post  
- Organization  
- CapabilityConfiguration  
- SystemsNode  
- Software  
- ResourceArtifact  
- System  
- DataElement |
| **SystemsElement** | Elements relating to system models. Specializations:  
- Resource  
- ResourceInteraction  
- ResourceMessage  
- ResourceInteraction  
- DataElement  
- ResourceStateMachine  
- FunctionEdge  
- Function |
| **UPDMElement** | A super type for all UPDM elements, providing a means of extending UPDM |
elements in a common way.
Specializations:
- All UPDM stereotypes
Quicklinks

The MDG Technology for UPDM makes use of Enterprise Architect's 'Quicklink' feature to make it quicker and easier to create correct and consistent UPDM models.

When you select an element, the Quicklink arrow displays next to the top-right corner of the element.

Drag the arrow away from the element and release it over empty diagram space. The Quicklink context menu displays, listing all the UPDM elements that could commonly be attached to the element, as shown.

Selecting the 'UPDM::Post | Commands' option in the context menu creates a new Post element connected to the Organization element by a Commands relationship.
Tagged Values for UPDM

UPDM is an extension of UML, which is extended by applying stereotypes to elements. The stereotypes in turn apply Tagged Values that provide additional information to that normally associated with a UML element. Since UPDM makes frequent use of Tagged Values, it is recommended to keep the Properties window docked and visible at all times, with the ‘UPDM’ section expanded.

Synchronize Tagged Values

The list of Tagged Values owned by an element can get out of date. A new version of the UML Profile might define new or modified Tagged Values for an element type, or as user might delete some. Also, you might apply the stereotype using the stereotype combo box, which doesn't add Tagged Values. If you want to refresh the list of Tagged Values for a single element, you can drag and drop the stereotype from the Diagram Toolbox onto the element and select the ‘Apply’ option. This only works for single diagram objects, and not for connectors.

If you want to refresh the list of Tagged Values for every element in your model, select the ‘Specialize > Technologies > UPDM > Synchronize Tagged Values’ menu option.

The URL/URI Tagged Value

In the UPDM Profile Specification the stereotype «UPDMElement» - from which all profile elements are derived - provides a Tagged Value URL/URI. In the MDG Technology for UPDM, this Tagged Value has been omitted and you must use the standard Enterprise Architect functionality to achieve the same result: that is, open the 'Properties' dialog for the element, select the 'Files' tab or page, and type in a web location.
Model Validation in MDG Technology for UPDM

The MDG Technology for UPDM provides model validation of UPDM models, validating and reporting errors against more than 160 different rules.

Configure Model Validation

Before being able to validate a model, you first have to select the rules to validate against. Select 'Design > Model > Manage > Validate > Configure Validation Rules' and deselect the checkbox against all validation rules except for the UPDM set.

Perform Model Validation

Open a diagram or select either a Package or a number of elements in the Browser window, then select the 'Design > Model > Manage > Validate > Validate Current Package' ribbon option (or press Ctrl+Alt+V). Validation results are displayed in the System Output window, which is opened if it isn't already displayed. To go to the element that caused a validation error, double-click on the error message in the System Output window.

Output

| MR8000005 - error (Linknote Driver (ActualPort)) | The object does not have a valid classifier (Permitted values: Port) |
| MR8000005 - error (Node1 (Node)) | The element has an invalid port (Required stereotypes: NodePort, RequiredPoint, ServicePoint) |
| MR8000005 - error (Transmitter (ResourcePort)) | The object does not have a valid type (Permitted values: ResourceInteractionItem) |
| MR8000005 - error (Anonymous (Node)) | The relationship does not have a valid end role (Permitted values: Node, Node) |
| MR8000005 - error (Anonymous (Node1)) | The relationship does not have a valid end type (Permitted values: Node) |
| MR8000005 - error (Node2 (Node)) | The element has an invalid port (Required stereotypes: NodePort, RequiredPoint, ServicePoint) |
| MR8000005 - error (Anonymous (OrganizationalExchange)) | The relationship does not have a valid information target (Permitted values: Node) |
| MR8000005 - error (Receiver (ResourcePort)) | The object does not have a valid type (Permitted values: ResourceInteractionItem) |
| MR8000005 - error (SendDistressSignal (OperationalActivityAction)) | The action has a missing or invalid activity (Required stereotypes: OperationsActivity) |
| MR8000005 - error (SendDistressSignal (OperationalActivityAction)) | The action has a missing or invalid behavior (Required stereotypes: OperationalActivity) |

Validation complete - 10 errors, 0 warnings

(c) Sparx Systems 2020
Model Validation Rules

Errors are indicated by an error code of the format MVRxxnnnn where:

- xx is 80 by default (if the MDG Technology for UPDM is the only Add-In that you have installed) but could be some other number, and
- nnnn is a hexadecimal number from 0001 to 0013 as described here

MVRxx0001 - activity

Error Message: The action has a missing or invalid activity (Required stereotype: <stereotypeList>)
The validation rule checks that stereotyped Action elements are owned by an Activity with the required stereotype.
Solution: Locate the Action in the Browser window, locate an Activity with one of the named stereotypes (or their specializations) or create a new one, and drag the Action to the Activity.

<table>
<thead>
<tr>
<th>Action Stereotypes</th>
<th>Activity Stereotypes</th>
</tr>
</thead>
<tbody>
<tr>
<td>FunctionAction</td>
<td>Function</td>
</tr>
<tr>
<td>OperationalActivityAction</td>
<td>OperationalActivity</td>
</tr>
<tr>
<td>ServiceOperationAction</td>
<td>Function</td>
</tr>
<tr>
<td>ServiceOperationAction</td>
<td>ServiceFunction</td>
</tr>
</tbody>
</table>

MVRxx0002 - annotatedElement

Error Message: The note has an invalid annotated element (Required stereotype: <stereotype>)
This validation rule checks that stereotyped Note elements are attached (by a NoteLink connector) to an element with the required stereotype.
Solution: Attach the Note to an element with the named stereotype (or one of its specializations). You can do this by either dragging the opposite end of the NoteLink connector, or deleting the NoteLink connector and creating a new one with the Quick Linker.

<table>
<thead>
<tr>
<th>Note Stereotypes</th>
<th>Annotated Element Stereotypes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alias</td>
<td>UPDMElement</td>
</tr>
<tr>
<td>ArchitectureMetadata</td>
<td>ArchitecturalDescription</td>
</tr>
<tr>
<td>Definition</td>
<td>UPDMElement</td>
</tr>
<tr>
<td>StandardConfiguration</td>
<td>CapabilityConfiguration</td>
</tr>
<tr>
<td>StereotypeExtension</td>
<td>UPDMElement</td>
</tr>
</tbody>
</table>
MVRxx0003 - behavior

Error Message: The action has a missing or invalid behavior (Required stereotype: <stereotype>)

This validation rule checks that stereotyped CallBehaviorAction elements call a Behavior with the required stereotype.

Solution: Right-click on the Action and select Advanced | Set Behavioral Classifier, or press Ctrl+L, and select a behavior element with the named stereotype (or one of its specializations).

<table>
<thead>
<tr>
<th>Action Stereotypes</th>
<th>Behavior Stereotypes</th>
</tr>
</thead>
<tbody>
<tr>
<td>OperationalActivityAction</td>
<td>OperationalActivity</td>
</tr>
<tr>
<td>ServiceFunctionAction</td>
<td>ServiceFunction</td>
</tr>
</tbody>
</table>

MVRxx0004 - class

Error Message: The object does not have a valid owning Class (Permitted values: <stereotypeList>)

This validation rule checks that stereotyped Property elements (Parts or attributes) are owned by a Class with the required stereotype.

Solution: Locate the property in the Browser window, locate a Class with one of the named stereotypes (or their specializations) or create a new one, and drag the property to the Class.

<table>
<thead>
<tr>
<th>Property Stereotypes</th>
<th>Class Stereotypes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equipment</td>
<td>OrganizationalResource</td>
</tr>
<tr>
<td>HostedSoftware</td>
<td>ResourceArtifact</td>
</tr>
<tr>
<td>HumanResource</td>
<td>CapabilityConfiguration</td>
</tr>
<tr>
<td>NodeChild</td>
<td>NodeParent</td>
</tr>
<tr>
<td>NodeRole</td>
<td>Node</td>
</tr>
<tr>
<td>Part</td>
<td>ResourceArtifact</td>
</tr>
<tr>
<td>PostRole</td>
<td>Organization</td>
</tr>
<tr>
<td>ProblemDomain</td>
<td>LogicalArchitecture</td>
</tr>
<tr>
<td>ProtocolLayer</td>
<td>Protocol</td>
</tr>
<tr>
<td>ResourceComponent</td>
<td>CapabilityConfiguration</td>
</tr>
<tr>
<td>ResourceRole</td>
<td>Resource</td>
</tr>
<tr>
<td>StructuralPart</td>
<td>EnterprisePhase</td>
</tr>
<tr>
<td>SubOrganization</td>
<td>Organization</td>
</tr>
</tbody>
</table>
TemporalPart | EnterprisePhase
---|---
UsedConfiguration | CapabilityConfiguration

**MVRxx0005 - classifier**

Error Message: The object does not have a valid classifier (Permitted value: `<stereotype>`)

This validation rule checks that stereotyped instance elements (objects) are classified by elements with the required stereotypes.

Solution: Select the object, right-click it and select Advanced | Instance Classifier, or press Ctrl+L, and select a classifier element with the named stereotype (or one of its specializations).

<table>
<thead>
<tr>
<th>Object Stereotypes</th>
<th>Classifier Stereotypes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActualMeasurementSet</td>
<td>MeasurementSet</td>
</tr>
<tr>
<td>ActualOrganization</td>
<td>Organization</td>
</tr>
<tr>
<td>ActualPerson</td>
<td>Person</td>
</tr>
<tr>
<td>ActualPost</td>
<td>Post</td>
</tr>
<tr>
<td>ActualProject</td>
<td>Project</td>
</tr>
<tr>
<td>ActualProjectMilestone</td>
<td>ProjectMilestoneType</td>
</tr>
<tr>
<td>FieldedCapability</td>
<td>CapabilityConfiguration</td>
</tr>
</tbody>
</table>

**MVRxx0006 - client**

Error Message: The relationship does not have a valid client (Permitted values: `<stereotypeList>`)

This validation rule checks that, for stereotyped Dependency or Realization relationships, their client (source) elements have the required stereotypes.

Solution: Drag the end of the relationship without the arrowhead to an element with the named stereotype (or one of its specializations).

<table>
<thead>
<tr>
<th>Relationship Stereotypes</th>
<th>Client Element Stereotypes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArbitraryRelationship</td>
<td>HighLevelOperationalConcept</td>
</tr>
<tr>
<td>ArchitecturalReference</td>
<td>ArchitecturalDescription</td>
</tr>
<tr>
<td>CompatibleWith</td>
<td>Node</td>
</tr>
<tr>
<td>DefinesArchitecture</td>
<td>ArchitecturalDescription</td>
</tr>
<tr>
<td>ExhibitsCapability</td>
<td>Node</td>
</tr>
</tbody>
</table>
MVRxx0007 - constrainedElement

Error Message: The constraint has an invalid constrained element (Required stereotype: %s)

This validation rule checks that stereotyped Constraint elements are attached (by a NoteLink) to elements with the required stereotypes.

Solution: Attach the constraint to an element with the named stereotype (or one of its specializations). You can do this by either dragging the opposite end of the NoteLink connector, or by deleting the NoteLink connector and creating a new one using the Quick Linker.

<table>
<thead>
<tr>
<th>Constraint Stereotypes</th>
<th>Constrained Element Stereotypes</th>
</tr>
</thead>
<tbody>
<tr>
<td>OperationalConstraint</td>
<td>SubjectOfOperationalConstraint</td>
</tr>
<tr>
<td>ResourceConstraint</td>
<td>SubjectOfResourceConstraint</td>
</tr>
</tbody>
</table>
MVRxx0008 - endRoles

Error Message: The relationship does not have a valid end role (Permitted values: <stereotypeList>)

This validation rule checks that, for stereotyped Association or Connector relationships, the elements at both ends of the relationship have the required stereotypes.

Solution: Drag one or both ends of the relationship to elements with the named stereotype (or one of its specializations).

<table>
<thead>
<tr>
<th>Relationship Stereotypes</th>
<th>End Element Stereotypes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Needline</td>
<td>NodeChild</td>
</tr>
<tr>
<td>Needline</td>
<td>NodePort</td>
</tr>
<tr>
<td>ResourceConnector</td>
<td>ResourcePort</td>
</tr>
<tr>
<td>ResourceInterface</td>
<td>ResourceRole</td>
</tr>
</tbody>
</table>

MVRxx0009 - endType

Error Message: The relationship does not have a valid end type (Permitted values: <stereotypeList>)

This validation rule checks that, for stereotyped connectors, the elements (Objects or Parts) at both ends of the relationship are typed by the required stereotypes.

Solution: Drag one or both ends of the relationship to elements that have types with the named stereotype (or one of its specializations).

<table>
<thead>
<tr>
<th>Connector Stereotypes</th>
<th>End Type Stereotypes</th>
</tr>
</thead>
<tbody>
<tr>
<td>EntityRelationship</td>
<td>EntityItem</td>
</tr>
<tr>
<td>Needline</td>
<td>Node</td>
</tr>
<tr>
<td>ResourceInterface</td>
<td>Resource</td>
</tr>
</tbody>
</table>

MVRxx000a - informationSource

Error Message: The relationship does not have a valid information source (Permitted values: <stereotypeList>)

This validation rule checks that stereotyped InformationFlow relationship source elements have the required stereotypes.

Solution: Drag the end of the information flow without the arrowhead to an element with the named stereotype (or one of its specializations).

<table>
<thead>
<tr>
<th>InformationFlow Stereotypes</th>
<th>Source Element Stereotypes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActualOrganizationRelation</td>
<td>ActualOrganizationalResource</td>
</tr>
<tr>
<td>Commands</td>
<td>OrganizationalResource</td>
</tr>
<tr>
<td>Controls</td>
<td>OrganizationalResource</td>
</tr>
<tr>
<td>----------</td>
<td>------------------------</td>
</tr>
<tr>
<td>OperationalExchange</td>
<td>Node</td>
</tr>
<tr>
<td>ResourceInteraction</td>
<td>Resource</td>
</tr>
</tbody>
</table>

**MVRxx000b - informationTarget**

Error Message: The relationship does not have a valid information target (Permitted values: `<stereotypeList>`)

This validation rule checks that stereotyped InformationFlow relationship target elements have the required stereotypes.

Solution: Drag the end of the information flow with the arrowhead to an element with the named stereotype (or one of its specializations).

<table>
<thead>
<tr>
<th>InformationFlow Stereotypes</th>
<th>Target Element Stereotypes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActualOrganizationRelationship</td>
<td>ActualOrganizationalResource</td>
</tr>
<tr>
<td>Commands</td>
<td>OrganizationalResource</td>
</tr>
<tr>
<td>Controls</td>
<td>OrganizationalResource</td>
</tr>
<tr>
<td>OperationalExchange</td>
<td>Node</td>
</tr>
<tr>
<td>ResourceInteraction</td>
<td>Resource</td>
</tr>
</tbody>
</table>

**MVRxx000c - ownedAttribute**

Error Message: The element has an invalid attribute (Required stereotype: `<stereotype>`)

This validation rule checks that, for stereotyped Class elements, any attributes that they own have the required stereotypes.

Solution: Replace the attribute with one with the named stereotype (or one of its specializations).

<table>
<thead>
<tr>
<th>Class Stereotypes</th>
<th>Attribute Stereotypes</th>
</tr>
</thead>
<tbody>
<tr>
<td>EntityItem</td>
<td>EntityAttribute</td>
</tr>
<tr>
<td>Environment</td>
<td>EnvironmentProperty</td>
</tr>
<tr>
<td>HighLevelOperationalConcept</td>
<td>ConceptRole</td>
</tr>
<tr>
<td>MeasurementSet</td>
<td>Measurement</td>
</tr>
<tr>
<td>ProjectMilestoneType</td>
<td>ProjectTheme</td>
</tr>
</tbody>
</table>
MVRxx000d - ownedOperation

Error Message: The element has an invalid operation (Required stereotype: %s)

This validation rule checks that, for stereotyped Class elements, any operations that they own have the required stereotypes.

Solution: Replace the operation with one with the named stereotype (or one of its specializations).

<table>
<thead>
<tr>
<th>Class Stereotype</th>
<th>Operation Stereotype</th>
</tr>
</thead>
<tbody>
<tr>
<td>ServiceInterface</td>
<td>ServiceOperation</td>
</tr>
</tbody>
</table>

MVRxx000e - ownedParameter

Error Message: The element has an invalid activity parameter (Required stereotype: %s)

This validation rule checks that, for stereotyped Activity elements, any ActivityParameter elements that they own have the required stereotypes.

Solution: Locate the ActivityParameter in the Browser window and drag and drop it onto an element with the appropriate stereotype, and/or replace the ActivityParameter in its current owner with an ActivityParameter with the named stereotype.

<table>
<thead>
<tr>
<th>Activity Stereotypes</th>
<th>ActivityParameter Stereotypes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function</td>
<td>FunctionParameter</td>
</tr>
<tr>
<td>OperationalActivity</td>
<td>OperationalParameter</td>
</tr>
</tbody>
</table>

MVRxx000f - ownedPort

Error Message: The element has an invalid Port (Required stereotypes: <stereotypeList>)

This validation rule checks that, for stereotyped Class elements, any Ports that they own have the required stereotypes.

Solution: Locate the Port in the Browser window and drag and drop it onto an element with the appropriate stereotype, and/or replace the Port in its current owner with a Port with one of the named stereotypes.

<table>
<thead>
<tr>
<th>Class Stereotypes</th>
<th>Port Stereotypes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node</td>
<td>NodePort</td>
</tr>
<tr>
<td>Node</td>
<td>RequestPoint</td>
</tr>
<tr>
<td>Node</td>
<td>ServicePoint</td>
</tr>
<tr>
<td>Resource</td>
<td>RequestPoint</td>
</tr>
</tbody>
</table>
MVRxx0010 - source

Error Message: The relationship does not have a valid source (Permitted values: <stereotypeList>)

This validation rule checks that stereotyped ActivityEdge connector source elements have the required stereotypes.

Solution: Drag the end of the relationship without the arrowhead to an element with the named stereotype (or one of its specializations).

<table>
<thead>
<tr>
<th>ActivityEdge Stereotypes</th>
<th>Source Element Stereotypes</th>
</tr>
</thead>
<tbody>
<tr>
<td>FunctionEdge</td>
<td>ServiceOperationAction</td>
</tr>
<tr>
<td>OperationalActivityEdge</td>
<td>OperationalActivityAction</td>
</tr>
</tbody>
</table>

MVRxx0011 - supplier

Error Message: The relationship does not have a valid supplier (Permitted values: <stereotypeList>)

This validation rule checks that stereotyped Dependency or Realization relationship supplier (target) elements have the required stereotypes.

Solution: Drag the end of the relationship with the arrowhead to an element with the named stereotype (or one of its specializations).

<table>
<thead>
<tr>
<th>Relationship Stereotypes</th>
<th>Supplier Element Stereotypes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArbitraryRelationship</td>
<td>HighLevelOperationalConcept</td>
</tr>
<tr>
<td>ArchitecturalReference</td>
<td>ArchitecturalDescription</td>
</tr>
<tr>
<td>CompatibleWith</td>
<td>ReferredLocation</td>
</tr>
<tr>
<td>DefinesArchitecture</td>
<td>EnterprisePhase</td>
</tr>
<tr>
<td>ExhibitsCapability</td>
<td>Capability</td>
</tr>
<tr>
<td>Expose</td>
<td>Capability</td>
</tr>
<tr>
<td>FillsPost</td>
<td>ActualPost</td>
</tr>
<tr>
<td>Forecast</td>
<td>SubjectOfForecast</td>
</tr>
<tr>
<td>ImplementsOperational</td>
<td>OperationalElement</td>
</tr>
<tr>
<td>MapsToCapability</td>
<td>Capability</td>
</tr>
</tbody>
</table>
### MVRxx0012 - target

Error Message: The relationship does not have a valid target (Permitted values: `<stereotypeList>`)

This validation rule checks that stereotyped `ActivityEdge` connector target elements have the required stereotypes.

Solution: Drag the end of the relationship with the arrowhead to an element with the named stereotype (or one of its specializations).

<table>
<thead>
<tr>
<th>ActivityEdge Stereotypes</th>
<th>Target Element Stereotypes</th>
</tr>
</thead>
<tbody>
<tr>
<td>FunctionEdge</td>
<td>ServiceOperationAction</td>
</tr>
<tr>
<td>OperationalActivityEdge</td>
<td>OperationalActivityAction</td>
</tr>
</tbody>
</table>

### MVRxx0013 - type

Error Message: The object does not have a valid type (Permitted value: `<stereotype>`)  

This validation rule checks that stereotyped `Property` elements (Parts or attributes) have type elements with the required stereotypes.

Solution: For Parts, right-click on the Part and select 'Advanced | Set Property Type', or press Ctrl+L, and select a type element with the named stereotype (or one of its specializations). For attributes, open the Features window for the attribute and select a type element with the named stereotype (or one of its specializations) in the 'Type' field.

<table>
<thead>
<tr>
<th>Property Stereotypes</th>
<th>Type Element Stereotypes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConceptRole</td>
<td>ConceptItem</td>
</tr>
<tr>
<td>EnvironmentProperty</td>
<td>EnvironmentalType</td>
</tr>
<tr>
<td>---------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>Equipment</td>
<td>ResourceArtifact</td>
</tr>
<tr>
<td>FunctionParameter</td>
<td>ResourceInteractionItem</td>
</tr>
<tr>
<td>HostedSoftware</td>
<td>Software</td>
</tr>
<tr>
<td>HumanResource</td>
<td>OrganizationalResource</td>
</tr>
<tr>
<td>KnownResource</td>
<td>Resource</td>
</tr>
<tr>
<td>NodePort</td>
<td>OperationalExchangeItem</td>
</tr>
<tr>
<td>NodeRole</td>
<td>Node</td>
</tr>
<tr>
<td>OperationalParameter</td>
<td>OperationalExchangeItem</td>
</tr>
<tr>
<td>Part</td>
<td>ResourceArtifact</td>
</tr>
<tr>
<td>PostRole</td>
<td>Post</td>
</tr>
<tr>
<td>ProjectTheme</td>
<td>ProjectThemeStatus</td>
</tr>
<tr>
<td>ProtocolLayer</td>
<td>Protocol</td>
</tr>
<tr>
<td>RequestPoint</td>
<td>ServiceInterface</td>
</tr>
<tr>
<td>ResourceComponent</td>
<td>ResourceArtifact</td>
</tr>
<tr>
<td>ResourcePort</td>
<td>ResourceInteractionItem</td>
</tr>
<tr>
<td>ServicePoint</td>
<td>ServiceInterface</td>
</tr>
<tr>
<td>StructuralPart</td>
<td>EnterprisePhase</td>
</tr>
<tr>
<td>SubOrganization</td>
<td>Organization</td>
</tr>
<tr>
<td>TemporalPart</td>
<td>EnterprisePhase</td>
</tr>
<tr>
<td>UsedConfiguration</td>
<td>CapabilityConfiguration</td>
</tr>
</tbody>
</table>
Model Views in MDG Technology for UPDM

The 'Model Views' tab of the Focus window displays a variety of different views on the model data, providing an alternative to the Browser window. The MDG Technology for UPDM uses this tab as a quick and easy method of locating all of your diagrams in the current model.

To open the 'Model Views' tab, select 'Start > Desktop > Design > Focus > Views'. Expand the appropriate folders and
double-click on the required diagram to open it.
Glossary

The MDG Technology for UPDM provides the ability to import descriptions of all UPDM stereotypes into the Enterprise Architect Glossary. This gives you a quick reference to the meaning of each stereotype, lists the views that the stereotype might appear in and, for abstract stereotypes, lists the concrete stereotypes that inherit from the abstract stereotype.

Import Glossary

You import the Glossary definitions into each model individually. To do this, select the 'Publish > Technologies > Import > Other Tools/Formats' ribbon option.

View the Glossary

To view the Glossary, select one of:

- 'Publish > Dictionary > Glossary > Glossary View to display the Project Glossary view
- 'Publish > Dictionary > Glossary > Edit' to open the 'Glossary' dialog
- In any dialog 'Notes' field, a Glossary hyperlink (underlined and colored blue)
Using Enterprise Architect Elements

Creating an instance from a Class

UPDM has Classifier/Instance pairs where the classifier describes a class of elements and the instance represents a single member of that Class. The Classifier/Instance pairs in UPDM are:

- MeasurementSet/ActualMeasurementSet
- Organization/ActualOrganization
- Person/ActualPerson
- Post/ActualPost
- Project/ActualProject
- ProjectMilestoneType/ActualProjectMilestone
- CapabilityConfiguration/FieldedCapability

If you have an element that is the classifier part of one of these Classifier/Instance pairs, you can choose between two main approaches for creating the instance:

1. Set the classifier of an existing instance - Click on the instance element in a diagram and then either press Ctrl+L or right-click and select 'Advanced | Instance Classifier'; the same command sets the type of a Port or Part.
2. Create an instance from an existing classifier - Press Ctrl while dragging the classifier element from the Browser window onto a diagram. The 'Paste Element' dialog displays; select the 'Paste as Instance of Element' option. An anonymous instance is created with the appropriate stereotype; select the instance, press F2 and give it a name.

Set the run state of an object

Where a classifier can own a set of attributes, an instance of that classifier can own a Slot for each attribute. The set of assigned values for these Slots is known as the run state. To set the run state of an object on a diagram, right-click on it and select 'Features | Set Run State' or press Ctrl+Shift+R.

Some stereotypes are defined by UPDM as extending the Slot metaclass. Each run state attribute represents a Slot, but it is not possible to stereotype Slots in Enterprise Architect, so UPDM's slot-extending stereotypes are not available in Enterprise Architect's implementation. UPDM stereotypes that extend Slot are:

- ActualMeasurement (ActualMeasurementSet)
- ActualOrganizationRole (ActualOrganization)
- MeasureOfPerformance (ActualMeasurementSet)
- ProjectStatus (ActualProjectMilestone)

Properties

Some stereotypes in UPDM are defined as extending the UML Property metaclass. This gives you the choice of a number of different representations for these elements in your model. If you drag one of the properties from the Toolbox onto a classifier element on a diagram, you are prompted to select to create an attribute, a Part, or a Port. These are all different representation of the UML Property metaclass; which one you choose depends on what rendering of the Property you want to see in your model.

Another representation of the UML Property metaclass is the Association End; to apply one of UPDM's Property stereotypes to an Association End:

1. Double-click on the element to display the 'Properties' dialog.
2. Select the 'Roles' tab.
3. Click on the [button next to the appropriate 'Stereotype' field.
4. On the 'Stereotype for Association' dialog, select 'UPDM' from the 'Profile' field.
5. Select every stereotype that applies.

Stereotypes that extend Property are:
- ConceptRole
- EntityAttribute
- EnvironmentProperty
- Equipment
- HostedSoftware
- HumanResource
- KnownResource
- Measurement
- NodeRole
- Part
- PerformanceParameter
- Platform
- PostRole
- ProblemDomain
- ProjectTheme
- ProtocolLayer
- ResourceComponent
- ServiceAttribute
- StructuralPart
- SubOrganization
- SubSystemPart
- TemporalPart
- UsedConfiguration
The Zachman Framework

The Zachman Framework is a widely used approach for engineering Enterprise Architecture. The Framework is a simple, logical structure that helps in organizing the information infrastructure of the Enterprise and provides many benefits in helping align technology with business needs.

Discussion

The topics described here provide an introduction to, and procedural explanation of, using the MDG Technology for the Zachman Framework in Enterprise Architect.

<table>
<thead>
<tr>
<th>Section</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>Welcome</td>
<td>This section provides an introduction to the Zachman Framework, and contains the formal documentation defining its use with Enterprise Architect.</td>
</tr>
<tr>
<td>Using the Zachman Framework</td>
<td>Get started with the Zachman Framework, learning about the model structure, templates, diagram types and more.</td>
</tr>
<tr>
<td>Model Validation</td>
<td>Learn how to develop and configure model validation for the Zachman Framework.</td>
</tr>
</tbody>
</table>
Welcome

Welcome to the Zachman Framework in Enterprise Architect.

Using this technology with Enterprise Architect, you can employ the Zachman Framework with the associated benefits of a powerful, open-standard modeling system. The Zachman Framework is already integrated with the Ultimate and Unified Editions; it can be purchased separately to be used with the Enterprise Architect Professional or Corporate Editions.

About the Zachman Framework

The Zachman Framework is a widely used approach for engineering Enterprise Architecture. The Framework is a simple, logical structure that helps in organizing the information infrastructure of the Enterprise.

While conceptually simple, the Zachman Framework provides many benefits in helping align technology with business needs. It has become a popular approach in defining Enterprise Architecture because it:

- Is platform neutral
- Is a powerful planning device
- Is both comprehensive and readily understood by non-technical people
- Assists in problem solving
- Helps in documenting enterprise-wide information system architecture


For further information, visit the Zachman Framework website.

Getting Started

For instructions on how to use the Zachman Framework, see the topics:

- Getting Started with the Zachman Framework and
- Using the Zachman Framework
Zachman Framework Copyright Notice

Copyright © 2007-2020 Sparx Systems Pty. Ltd. All rights reserved.

The MDG Technology for Zachman Framework software contains proprietary information of Sparx Systems Pty Ltd. It is provided under a license agreement containing restrictions on use and disclosure and is also protected by copyright law. Reverse engineering of the software is prohibited. Please read the product license agreement for full details.

Due to continued product development, this information may change without notice. The information and intellectual property contained herein is confidential between Sparx Systems and the client and remains the exclusive property of Sparx Systems. If you find any problems in the documentation, please report them to us in writing. Sparx Systems does not warrant that this document is error-free. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise without the prior written permission of Sparx Systems. Licensed users are granted the right to print a single hardcopy of the user manual per licensed copy of the software, but may not sell, distribute or otherwise dispose of the hardcopy without written consent of Sparx Systems.

Sparx Systems Pty. Ltd.
99 Albert St,
Creswick, Victoria 3363,
AUSTRALIA
Phone: +61 (3) 5345 1140
Fax: +61 (3) 5345 1104
Support Email: support@sparxsystems.com
Sales Email: sales@sparxsystems.com
Website: sparxsystems.com
MDG Technology for Zachman Framework Software Product License Agreement

This Software Product License Agreement relates to the separately-purchased MDG Technology for Zachman Framework for use with the Professional and Corporate Editions of Sparx Systems Enterprise Architect. The MDG Technology integrated with the Unified and Ultimate Editions of Enterprise Architect is subject to the Sparx Systems Enterprise Architect Modelling Tool.

MDG Technology for Zachman Framework - Enterprise Architect MDG Add-In, Version 1.1
Copyright © 2007-2020 Sparx Systems Pty Ltd. All Rights Reserved

IMPORTANT-READ CAREFULLY: This End User License Agreement ("EULA") is a legal agreement between YOU as Licensee and SPARX for the SOFTWARE PRODUCT identified above. By installing, copying, or otherwise using the SOFTWARE PRODUCT, YOU agree to be bound by the terms of this EULA. If YOU do not agree to the terms of this EULA, promptly delete the unused SOFTWARE PRODUCT.

The copyright in the SOFTWARE PRODUCT and its documentation is owned by Sparx Systems Pty Ltd, A.B.N 38 085 034 546. Subject to the terms of this EULA, YOU are granted a non-exclusive right for the duration of the EULA to use the SOFTWARE PRODUCT. YOU do not acquire ownership of copyright or other intellectual property rights in any part of the SOFTWARE PRODUCT by virtue of this EULA.

Your use of this software indicates your acceptance of this EULA and warranty.

DEFINITIONS

In this End User License Agreement, unless the contrary intention appears,

- "EULA" means this End User License Agreement
- "SPARX" means Sparx Systems Pty Ltd A.C.N 085 034 546
- "Licensee" means YOU, or the organization (if any) on whose behalf YOU are taking the EULA
- "SOFTWARE PRODUCT" or "SOFTWARE" means MDG Technology for Zachman Framework, which includes computer software and associated media and printed materials, and may include online or electronic documentation
- "Support Services" means email-based support provided by SPARX, including advice on usage of Enterprise Architect, investigation of bugs, fixes, repairs of models, if and when appropriate, and general product support
- "SPARX support engineers" means employees of SPARX who provide on-line support services

GRANT OF LICENSE

In accordance with the terms of this EULA YOU are granted the following rights:

- To install and use ONE copy of the SOFTWARE PRODUCT or, in its place, any prior version for the same operating system, on a single computer; as the primary user of the computer on which the SOFTWARE PRODUCT is installed, YOU may make a second copy for your exclusive use on either a home or portable computer
- To store or install a copy of the SOFTWARE PRODUCT on a storage device, such as a network server, used only to install or run the SOFTWARE PRODUCT over an internal network
- To make copies of the SOFTWARE PRODUCT for backup, archival and instructional purposes
EVALUATION LICENSE

The Trial Edition of MDG Technology for Zachman Framework is not free software. Subject to the terms of this agreement, YOU are hereby licensed to use this software for evaluation purposes without charge for a period of thirty (30) days.

Upon expiration of the thirty (30) days, the SOFTWARE PRODUCT must be removed from the computer. Unregistered use of MDG Technology for Zachman Framework after the 30-day evaluation period is in violation of Australian, U.S. and international copyright laws.

SPARX may extend the evaluation period on request and at their discretion.

If YOU choose to use this software after the 30 day evaluation period a license must be purchased (as described at https://sparxsystems.com/products/mdg/tech/zachman/purchase.html). Upon payment of the license fee, YOU will be sent details on where to download the registered edition of MDG Technology for Zachman Framework and will be provided with a suitable software 'key' by email.

ADDITIONAL RIGHTS AND LIMITATIONS

YOU hereby undertake not to sell or sub-licence the SOFTWARE PRODUCT other than as expressly authorized by this EULA.

NO WARRANTY. The SOFTWARE PRODUCT is provided "AS IS", without warranty of any kind, and SPARX expressly disclaims all warranties and/or conditions with respect to the SOFTWARE PRODUCT, either express, implied or statutory, including, but not limited to, the implied warranties and/or conditions of merchantability, of satisfactory quality, of fitness for a particular purpose, of accuracy, of quiet enjoyment, and of non-infringement of third party rights.

LIMITATION

Under no circumstances shall SPARX be liable for any incidental, special, indirect or consequential damages arising out of or relating to this license or YOUR use, reproduction, modification, distribution of the SOFTWARE PRODUCT, or any portion thereof, whether under a theory of contract, warranty, strict liability or otherwise, even if the copyright holder has been advised of the possibility of such damages and notwithstanding the failure of essential purpose of any remedy.

TRADEMARKS

All names of products and companies used in this EULA, the SOFTWARE PRODUCT, or the enclosed documentation can be trademarks of their corresponding owners. Their use in this EULA is intended to be in compliance with the respective guidelines and licenses. Windows, Windows XP, Windows 2003 Server, Windows 2008 Server, Windows Vista, Windows 7 and Windows 8 are trademarks of Microsoft.

The Zachman Framework for Enterprise Architecture™ is a trademark of John A. Zachman and Zachman International.

GOVERNING LAW

This agreement shall be construed in accordance with the laws of the Commonwealth of AUSTRALIA, in the state of Victoria.
Acknowledgement of Trademarks

Sparx Systems acknowledge these trademarks, which are used throughout the MDG for Zachman Framework documentation.

Trademarks of Microsoft

- Microsoft Word
- Microsoft Office
- Windows®

Trademarks of the Object Management Group

- Object Management Group™
- OMG™
- UML™
- Unified Modeling Language™

Trademark of John A. Zachman and Zachman International

- The Zachman Framework For Enterprise Architecture™
Support for the Zachman Framework

Technical support for the Zachman Framework is available to registered users of Enterprise Architect through the same channels as for Enterprise Architect itself.
Zachman Framework System Requirements

Zachman Framework version 1.1.4 runs under the environments identified here.

**Microsoft® Operating Systems Supported**

- Windows 8
- Windows 7
- Windows Vista
- Windows 2008 Server
- Windows 2003 Server
- Windows XP Service Pack 2

**Enterprise Architect Versions Supported**

- Enterprise Architect Version 7.1 or later

**Notes**

- 32 bit and 64 bit operating systems supported
Getting Started with the Zachman Framework

When you install the Unified or Ultimate Edition of Enterprise Architect, the Zachman Framework is fully enabled and ready to use.

If you have the Corporate or Professional Edition of Enterprise Architect, you can purchase and install the MDG Technology for Zachman Framework separately; once you have entered the registration key for the MDG Technology for Zachman Framework, it is automatically available in and integrated with Enterprise Architect, as for the Unified and Ultimate Editions.

Access the MDG Technology For Zachman Framework

1. Create a new Enterprise Architect project file, and click on the top-level Package.
2. Select the 'Design > Model > Perspectives > Model Wizard' option.
3. In the Model Wizard, select the 'Enterprise Architecture > Zachman' Perspective and the 'Zachman Framework' Pattern.
4. Click on the Create Patterns button.

A new base Zachman model is created in the Browser window, containing the Zachman Framework diagram and the Planner, Owner, Designer, Builder, Subcontractor and Functioning Enterprise Packages.
Using the Zachman Framework

The Zachman Framework provides a model-based framework for planning, designing and implementing the Architecture for an Enterprise. The starter model provided with the Technology acts as a base upon which you can build the Enterprise Architecture. You can create the appropriate diagrams from the extended Enterprise Architect UML diagram set, using Toolbox pages that support every cell of the Zachman classification framework.

The Technology also provides model validation and reporting capabilities for strategic project plans.

Within Enterprise Architect you can choose between Diagram View and Element List View. Element List View can be used in cells where you prefer to define only the model artifacts.

You can also align cells across the framework (horizontally and vertically) through the Enterprise Architect Relationship Matrix.

You can view a demonstration video of the MDG Technology For Zachman Framework in use, on the Sparx Systems website.

The Zachman Framework Help topics provide a detailed exploration of the Zachman Framework tools and features, such as:

- The example Enterprise Architect model for the Zachman Framework
- UML profiles (Toolbox pages) for use within specific Zachman Framework cells
- A diagram interface for the Zachman Framework
- New diagram types specific to the Zachman Framework
- A flexible model starter-structure
- Report generation capabilities for strategic project plans

The MDG Technology For Zachman Framework is integrated with the features of Enterprise Architect.
The Zachman Framework Interface Diagram

The Zachman Framework is a predefined model in Enterprise Architect. The model-level diagram of the model structure is the Zachman Framework Interface diagram, which serves as a template for the development of Enterprise Architecture based on the Zachman classification framework.

Each cell links to the relevant Zachman Framework diagram in the child Packages in the base model.
Zachman Framework Model Structure

The Zachman Framework provides a Framework model template, in which each Zachman Perspective (or row) is modeled as the highest-level Package inside the model. Cells belonging to the Perspectives are modeled as child Packages of the appropriate row Package.
The Zachman Framework Model Template

The Zachman Framework Model Template provides the model skeleton from which you can develop your Enterprise definition.

Add a new Zachman Framework model to the project

1. Right-click on the root node and select 'Add a Model using Wizard'. The 'Model Wizard' view displays.
2. On the 'Model Patterns' tab, click on the <name> Perspective button and select 'Enterprise Architecture > Zachman' from the list.
3. Select the 'Zachman Framework' pattern.
4. Click on the Create Patterns button.
Zachman Framework Diagrams

The Zachman Framework introduces new diagram types that support modeling of the Zachman Classification Framework. A Zachman Framework diagram is created in the same way as any other diagram in Enterprise Architect. The Technology provides access to these categories of diagram through the 'New Diagram' dialog:

- Planner
- Owner
- Designer
- Builder
- Subcontractor
- Zachman Framework Interface
Zachman Framework Diagram Types

The Zachman Framework further extends the Enterprise Architect diagram set to support the Framework, with diagram types appropriate to each cell of the Zachman Framework.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Planner (Objective/Scope)</td>
<td>Business Data</td>
<td>High Level Business Process</td>
<td>Business Locations</td>
<td>Organization Chart</td>
<td>Business Events</td>
<td>Business Motivation</td>
</tr>
<tr>
<td>Owner (Conceptual)</td>
<td>Data Map</td>
<td>Process Analysis</td>
<td>Business Logistics</td>
<td>BPMN</td>
<td>Event Schedule</td>
<td>Strategy Map</td>
</tr>
<tr>
<td></td>
<td>Add In Generated Process Map</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Mind Mapping</td>
</tr>
<tr>
<td>Designer (Logical)</td>
<td>Class - (Platform Independent Model)</td>
<td>Activity</td>
<td>Data Distribution Architecture</td>
<td>Use Case</td>
<td>State Transition</td>
<td>Business Rule Model</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Requirements</td>
</tr>
<tr>
<td>Builder (Physical)</td>
<td>Physical Data Model</td>
<td>Class - (Platform Specific Model) Component</td>
<td>Deployment</td>
<td>User Interface</td>
<td>Interaction Communication</td>
<td>Rule Design</td>
</tr>
</tbody>
</table>

**Legend:**
- UML Diagrams
- UML Profile for Zachman Framework
- Enterprise Architect extension
The Zachman Framework Toolbox

The Zachman Framework pages of the Diagram Toolbox provide elements and relationships for all the Zachman Framework diagrams that the MDG Technology supports. The Zachman Framework Toolbox pages can be accessed by clicking on ![Zachman](image) and specifying 'Zachman' in the 'Find Toolbox Item' dialog. The Diagram Toolbox can be docked on either side of the diagram, or free floated on top of the diagram to expose more surface for editing.

Diagrams for Toolboxes

This table shows, for each Zachman Framework cell, the diagram that could be used.

<table>
<thead>
<tr>
<th>Zachman Cell</th>
<th>Diagram</th>
</tr>
</thead>
<tbody>
<tr>
<td>Planner - Data</td>
<td>Business Data</td>
</tr>
<tr>
<td>Planner - Function</td>
<td>Business Process</td>
</tr>
<tr>
<td>Planner - Location</td>
<td>Business Locations</td>
</tr>
<tr>
<td>Planner - People</td>
<td>Organization Chart</td>
</tr>
<tr>
<td>Planner - Timing</td>
<td>Business Events</td>
</tr>
<tr>
<td>Planner - Motivation</td>
<td>Business Motivation</td>
</tr>
<tr>
<td>Owner - Data</td>
<td>Data Map and Process Map (Generated by Add-In)</td>
</tr>
<tr>
<td>Owner - Function</td>
<td>Process Analysis</td>
</tr>
<tr>
<td>Owner - Location</td>
<td>Business Logistics</td>
</tr>
<tr>
<td>Owner - People</td>
<td>BPMN</td>
</tr>
<tr>
<td>Owner - Timing</td>
<td>Event Schedule</td>
</tr>
<tr>
<td>Owner - Motivation</td>
<td>Enterprise Architect Mind Mapping diagram and Strategy Map</td>
</tr>
<tr>
<td>Designer - Data</td>
<td>Class</td>
</tr>
<tr>
<td>Designer - Function</td>
<td>Activity</td>
</tr>
<tr>
<td>Designer - Location</td>
<td>Data Distribution Architecture</td>
</tr>
<tr>
<td>Designer - People</td>
<td>Use Case</td>
</tr>
<tr>
<td>Designer - Timing</td>
<td>State Transition</td>
</tr>
<tr>
<td>Designer - Motivation</td>
<td>Business Rule Model</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>Builder - Data</td>
<td>Physical Data Model</td>
</tr>
<tr>
<td>Builder - Function</td>
<td>Class and Component</td>
</tr>
<tr>
<td>Builder - Location</td>
<td>Deployment</td>
</tr>
<tr>
<td>Builder - People</td>
<td>User Interface</td>
</tr>
<tr>
<td>Builder - Timing</td>
<td>Communication and Interaction</td>
</tr>
<tr>
<td>Builder - Motivation</td>
<td>Rule Design</td>
</tr>
<tr>
<td>Subcontractor - Data</td>
<td>Data Definition; default toolbox for the diagram is Custom.</td>
</tr>
<tr>
<td>Subcontractor – Function</td>
<td>No diagram defined – Code generation is done in this cell.</td>
</tr>
<tr>
<td>Subcontractor - Location</td>
<td>Network Architecture</td>
</tr>
<tr>
<td>Subcontractor - People</td>
<td>Security Architecture</td>
</tr>
<tr>
<td>Subcontractor - Timing</td>
<td>Timing</td>
</tr>
<tr>
<td>Subcontractor - Motivation</td>
<td>Rule Specification</td>
</tr>
</tbody>
</table>
Business Data Page

Business Data Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actor</td>
<td>Models a stakeholder or any other human resource of the enterprise.</td>
</tr>
<tr>
<td>Asset</td>
<td>Represents the enterprise resources that could be estimated for value.</td>
</tr>
<tr>
<td>Business Entity</td>
<td>Represents generic enterprise resources.</td>
</tr>
<tr>
<td>Document Asset</td>
<td>A subtype of Asset that captures the important documents of the enterprise.</td>
</tr>
<tr>
<td>Equipment Asset</td>
<td>A subtype of Asset that captures the equipment resources of the enterprise.</td>
</tr>
<tr>
<td>Assumption</td>
<td>Captures the assumptions made in information manipulation.</td>
</tr>
<tr>
<td>Principle</td>
<td>Defines the Principles framed and followed in the enterprise.</td>
</tr>
<tr>
<td>Standard</td>
<td>Defines the standards followed in the Enterprise.</td>
</tr>
</tbody>
</table>

Notes

- Elements and connectors common to Enterprise Architect UML and Extended diagrams are documented in the Object Toolbox section
Business Process Pages

Business Process Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actor</td>
<td>Models a stakeholder or any other human resource of the Enterprise.</td>
</tr>
<tr>
<td>Decision</td>
<td>Indicates the point of conditional progression where a business decision is taken.</td>
</tr>
<tr>
<td>Business Function</td>
<td>Represents a major function performed by the enterprise or a part of the enterprise.</td>
</tr>
<tr>
<td>Business Process</td>
<td>Represents a function or behavior of the enterprise or part of the enterprise.</td>
</tr>
<tr>
<td>Business Entity</td>
<td>Represents generic enterprise resources.</td>
</tr>
<tr>
<td>Invokes</td>
<td>A relationship that defines the invocation of a business process.</td>
</tr>
</tbody>
</table>

Notes

- Elements and connectors common to Enterprise Architect UML and Extended diagrams are documented in the Object Toolbox section.
Business Location Page

Business Location Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Branch Office</td>
<td>Models a Business Location as a Branch Office.</td>
</tr>
<tr>
<td>Client Place</td>
<td>Models a Business Location as a Client Place.</td>
</tr>
<tr>
<td>Head Quarters</td>
<td>Models a Business Location as a Head Quarters.</td>
</tr>
<tr>
<td>Business Location</td>
<td>Models the location from which the business operates.</td>
</tr>
<tr>
<td>Office Block</td>
<td>Models a Business Location as an Office Block.</td>
</tr>
<tr>
<td>Sales Agent</td>
<td>Models a Business Location as a Sales Agent.</td>
</tr>
<tr>
<td>Supplier</td>
<td>Models a Business Location as a Supplier.</td>
</tr>
</tbody>
</table>

Notes

- Elements and connectors common to Enterprise Architect UML and Extended diagrams are documented in the [Object Toolbox](#) section.
Business Motivation Pages

Business Motivation Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Goal</td>
<td>Models what is to be achieved by the enterprise, with specifications defined by the Tagged Values.</td>
</tr>
<tr>
<td>Mission</td>
<td>Models the mission statement, policies and values of the enterprise.</td>
</tr>
<tr>
<td>Strategy</td>
<td>Models the strategy statements for the business plan.</td>
</tr>
<tr>
<td>Assumption</td>
<td>Models the assumptions made in information manipulation. Tagged Value Type = Enterprise / Business / System / Application / Technology / Data.</td>
</tr>
<tr>
<td>Standard</td>
<td>Defines the standards followed in the enterprise. Tagged Value Type = Enterprise / Business / System / Application / Technology / Data.</td>
</tr>
</tbody>
</table>

Notes

- Elements and connectors common to Enterprise Architect UML and Extended diagrams are documented in the Object Toolbox section
Organization Chart Pages

Organization Chart Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Board of Directors</td>
<td>Captures the details of the board of directors.</td>
</tr>
<tr>
<td>StakeHolder</td>
<td>Defines a stakeholder of the enterprise.</td>
</tr>
<tr>
<td>External Organization</td>
<td>Defines any external business unit that is not under direct control of the enterprise, but has a relationship with the enterprise.</td>
</tr>
<tr>
<td>Organization Unit</td>
<td>Defines any business unit that is under direct control of the enterprise.</td>
</tr>
<tr>
<td>Personnel</td>
<td>Captures the details of personnel in an enterprise.</td>
</tr>
<tr>
<td>In Contract</td>
<td>A connector that represents the contract-based relationships between business units.</td>
</tr>
<tr>
<td>Works For</td>
<td>A connector that captures the details of team links; for example, Stakeholder 1 works for Organization Unit 1.</td>
</tr>
<tr>
<td>Supervise</td>
<td>A connector that captures process supervision details.</td>
</tr>
<tr>
<td>Control</td>
<td>A connector that captures Unit in charge or Person in charge information.</td>
</tr>
</tbody>
</table>

Notes

- Elements and connectors common to Enterprise Architect UML and Extended diagrams are documented in the [Object Toolbox](#) section
Business Events Pages

Business Event Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business Event</td>
<td>Captures major business events of the enterprise.</td>
</tr>
<tr>
<td>Trigger</td>
<td>Indicates that a Business Event triggers another event or a business process.</td>
</tr>
</tbody>
</table>
Data Map Pages

Data Map Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Principal Entity</td>
<td>Represents a business entity that forms a resource of the enterprise.</td>
</tr>
<tr>
<td>Intersecting Entity</td>
<td>Normalizes the many-to-many relationship between principal entities.</td>
</tr>
<tr>
<td>Structure Entity</td>
<td>Captures potential knowledge-based entities.</td>
</tr>
</tbody>
</table>

Notes

- Elements and connectors common to Enterprise Architect UML and Extended diagrams are documented in the [Object Toolbox](#) section
Business Logistics Pages

Business Logistics Items and Relations

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Branch Office</td>
<td>Models a Business Location as a Branch Office.</td>
</tr>
<tr>
<td>Client Place</td>
<td>Models a Business Location as a Client location</td>
</tr>
<tr>
<td>Head Quarters</td>
<td>Models a Business Location as Head Quarters.</td>
</tr>
<tr>
<td>Business Location</td>
<td>Models the location from which the business operates.</td>
</tr>
<tr>
<td>Office Block</td>
<td>Models a Business Location as an Office Block.</td>
</tr>
<tr>
<td>Sales Agent</td>
<td>Models a Business Location as a Sales Agent.</td>
</tr>
<tr>
<td>Supplier</td>
<td>Models a Business Location as a Supplier.</td>
</tr>
<tr>
<td>Communicate</td>
<td>Indicates that a business location communicates directly with another business location.</td>
</tr>
<tr>
<td>Internet</td>
<td>Indicates that the means of communication is the World Wide Web.</td>
</tr>
<tr>
<td>Snail Mail</td>
<td>Indicates that the means of communication is the postal system or courier services.</td>
</tr>
<tr>
<td>Phone</td>
<td>Indicates that the means of communication is the telephone.</td>
</tr>
<tr>
<td>In Person</td>
<td>Indicates that the means of communication is direct person-to-person.</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------------------------------------------------------------------</td>
</tr>
<tr>
<td>Intranet</td>
<td>Indicates that the means of communication is the local intranet or WAN.</td>
</tr>
</tbody>
</table>

**Notes**

- Elements and connectors common to Enterprise Architect UML and Extended diagrams are documented in the [Object Toolbox](#) section
BPMN Pages

The BPMN Toolbox pages provide the graphical (Core) and non-graphical (Types) Business Process Model and Notation (BPMN) elements for use on Business Process diagrams through the Zachman Framework Technology. Specifications of these elements and relationships are defined by Tagged Values.

BPMN Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business Process</td>
<td>Defines a business process; an extension of a composite Activity.</td>
</tr>
<tr>
<td>Activity</td>
<td>Defines an activity within a business process.</td>
</tr>
<tr>
<td>Start Event</td>
<td>Defines the initiating event in a process.</td>
</tr>
<tr>
<td>Intermediate Event</td>
<td>Defines an intermediate event in a process.</td>
</tr>
<tr>
<td>End Event</td>
<td>Defines the terminating event in a process.</td>
</tr>
<tr>
<td>Gateway</td>
<td>Defines a decision point in a business process. If a condition is true, then processing continues one way; if not, then another.</td>
</tr>
<tr>
<td>Pool</td>
<td>Logically organizes an Activity; an extension of a Partition element.</td>
</tr>
<tr>
<td>----------------------</td>
<td>---------------------------------------------------------------------</td>
</tr>
<tr>
<td>Lane</td>
<td>Subdivides a Pool; an extension of a Partition element.</td>
</tr>
<tr>
<td>Data Object</td>
<td>Defines a physical piece of information used or produced by a system; an extension of an Artifact element.</td>
</tr>
<tr>
<td>Group</td>
<td>Groups a number of other elements; an extension of a Boundary element.</td>
</tr>
<tr>
<td>Text Annotation</td>
<td>A comment.</td>
</tr>
<tr>
<td>Sequence Flow</td>
<td>Defines the flow of an activity; an extension of a Control Flow relationship.</td>
</tr>
<tr>
<td>Message Flow</td>
<td>Defines the flow of communications in a process; an extension of a Control Flow relationship.</td>
</tr>
<tr>
<td>Association</td>
<td>Associates information and artifacts with flow objects.</td>
</tr>
<tr>
<td>Message</td>
<td>Defines a message; an extension of a Class element.</td>
</tr>
<tr>
<td>Participant</td>
<td>Defines a participant in an activity; an extension of a Class element.</td>
</tr>
<tr>
<td>Rule</td>
<td>Defines business rule statements; an extension of a Class element.</td>
</tr>
<tr>
<td>Transaction</td>
<td>Defines a transaction in an activity; an extension of a Class element.</td>
</tr>
<tr>
<td>Web Service</td>
<td>Defines a web service; an extension of a Class element.</td>
</tr>
<tr>
<td>Property</td>
<td>Assigns a property to an element; an extension of an attribute.</td>
</tr>
</tbody>
</table>

**Notes**

- Enterprise Architect is delivered with the BPMN Technologies (for BPMN 1.0, 1.1 and 2.0) automatically installed, providing BPMN profiles and Toolboxes separate from this Zachman version; to make even further use of BPMN facilities, download the BPMN Add-In from:

  https://sparxsystems.com/products/mdg_bpmn.html
Event Schedule Pages

Event Schedule Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business Event</td>
<td>Captures major business events of the enterprise.</td>
</tr>
<tr>
<td>Business Cycle</td>
<td>Captures major business cycles of the enterprise.</td>
</tr>
<tr>
<td>Event Node</td>
<td>Captures the event points in a business cycle.</td>
</tr>
</tbody>
</table>

Notes

- Elements and connectors common to Enterprise Architect UML and Extended diagrams are documented in the Object Toolbox section.
Strategy Map Pages

Strategy Map Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strategy</td>
<td>Captures the strategy statements for the business plan.</td>
</tr>
<tr>
<td>Goal</td>
<td>Captures what is to be achieved by the enterprise, with specifications defined by the Tagged Values.</td>
</tr>
<tr>
<td>Business Perspective</td>
<td>Relates the strategies to a specific category.</td>
</tr>
<tr>
<td>Strategy Link</td>
<td>Indicates that a strategy is linked to another strategy or goal.</td>
</tr>
</tbody>
</table>
Data Distribution Architecture Pages

Data Distribution Architecture Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>File</td>
<td>Represents a file.</td>
</tr>
<tr>
<td>Desktop</td>
<td>Represents a desktop.</td>
</tr>
<tr>
<td>Processor</td>
<td>Represents a processor.</td>
</tr>
<tr>
<td>Registry</td>
<td>Represents a registry.</td>
</tr>
<tr>
<td>Storage Device</td>
<td>Represents a storage device.</td>
</tr>
<tr>
<td>Web Service</td>
<td>Represents a web service.</td>
</tr>
</tbody>
</table>

Notes

- Elements and connectors common to Enterprise Architect UML and Extended diagrams are documented in the [Object Toolbox](#) section
Business Rule Model Pages

Business Rule Model Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business Rule</td>
<td>Captures the Business Rule statements.</td>
</tr>
<tr>
<td>Principle</td>
<td>Defines the Principles framed and followed in the Enterprise.</td>
</tr>
<tr>
<td></td>
<td>Tag Value Type = Enterprise / Business / System / Application / Technology / Data.</td>
</tr>
<tr>
<td>Standard</td>
<td>Defines the standards followed in the Enterprise.</td>
</tr>
<tr>
<td></td>
<td>Tag Value Type = Enterprise / Business / System / Application / Technology / Data.</td>
</tr>
<tr>
<td>Assumption</td>
<td>Captures the assumptions made in information manipulation.</td>
</tr>
<tr>
<td></td>
<td>Tag Value Type = Enterprise / Business / System / Application / Technology / Data.</td>
</tr>
<tr>
<td>Based On</td>
<td>Indicates that a rule is based on another model element, which forms the rationale for the rule.</td>
</tr>
<tr>
<td>Replaces</td>
<td>Indicates that a new rule replaces another rule.</td>
</tr>
<tr>
<td>Conflict</td>
<td>Indicates that a rule conflicts with another defined rule.</td>
</tr>
<tr>
<td>Equivalent To</td>
<td>Indicates that a rule is equivalent to another rule.</td>
</tr>
<tr>
<td>Exception To</td>
<td>Indicates exceptions for a rule.</td>
</tr>
</tbody>
</table>
Notes

- Elements and connectors common to Enterprise Architect UML and Extended diagrams are documented in the [Object Toolbox](#) section.
Rule Design Pages

### Rule Design Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Formal Rule</td>
<td>Represents a business rule transformed to a technology-specific logical rule or constraint statement.</td>
</tr>
</tbody>
</table>
| Principle | Defines the Principles framed and followed in the Enterprise.  
Tag Value Type = Enterprise / Business / System / Application / Technology / Data. |
| Standard | Used to define the Standards followed in the Enterprise.  
Tag Value Type = Enterprise / Business / System / Application / Technology / Data. |
| Assumption | Used to capture the assumptions made in information manipulation.  
Tag Value Type = Enterprise / Business / System / Application / Technology / Data. |
| Applied To | Indicates that a Formal Rule is applied to other model artifacts such as Scenarios or Activities. |
| Validates | Indicates that a model artifact validates a Formal Rule. |

**Notes**

- Elements and connectors common to Enterprise Architect UML and Extended diagrams are documented in the [Object Toolbox](#) section
## Network Architecture Pages

### Network Architecture Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Artifact</td>
<td>Generic graphical element used to capture information.</td>
</tr>
<tr>
<td>Document Artifact</td>
<td>Generic graphical element used to capture detailed information such as network configuration details.</td>
</tr>
</tbody>
</table>

### Notes

- For a full description of Artifact elements, see the Artifact topic
Rule Specification Pages

Rule Specification Toolbox

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rule Impl Component</td>
<td>Captures the component implementing a rule.</td>
</tr>
<tr>
<td>Principle</td>
<td>Defines the Principles framed and followed in the enterprise.</td>
</tr>
<tr>
<td></td>
<td>Tag Value Type = Enterprise / Business / System / Application / Technology /</td>
</tr>
<tr>
<td></td>
<td>Data.</td>
</tr>
<tr>
<td>Standard</td>
<td>Defines the Standards followed in the enterprise.</td>
</tr>
<tr>
<td></td>
<td>Tag Value Type = Enterprise / Business / System / Application / Technology /</td>
</tr>
<tr>
<td></td>
<td>Data.</td>
</tr>
<tr>
<td>Assumption</td>
<td>Captures the assumptions made in information manipulation.</td>
</tr>
<tr>
<td></td>
<td>Tag Value Type = Enterprise / Business / System / Application / Technology /</td>
</tr>
<tr>
<td></td>
<td>Data.</td>
</tr>
<tr>
<td>Implements</td>
<td>Indicates that a Rule Impl Component implements a rule.</td>
</tr>
<tr>
<td>Violates</td>
<td>Indicates that the rule is violated by the connecting model element.</td>
</tr>
</tbody>
</table>
Tagged Values for Zachman Framework

The Zachman Framework makes extensive use of Tagged Values to assign custom properties to the various Zachman Framework elements. When creating or viewing a Zachman Framework model, it is recommended that you keep the Properties window docked and visible at all times, with the 'ZF' section expanded.

Access

| Ribbon | Start > Desktop > Design > Properties > Tags  
|        | Explore > Portals > Windows > Properties > Tagged Values |
|        | Keyboard Shortcuts  
|        | Ctrl+2 |

Synchronize Tagged Values

From time to time you might need to add missing Tagged Values to all elements in the model that require them, such as:

- Whenever you create a new element by any means other than directly dropping the element from the Zachman Framework Toolbox pages
- Before using a new version of the Technology, to update the Tagged Values of elements in existing models to the latest version of the Zachman Framework profile

You can do this using the 'Synchronize Stereotype' option on the icons in the Zachman Framework pages of the Diagram Toolbox.
Data Map Analysis

A valid Data Map diagram is basically an Entity Relationship diagram constructed using Principal Entity, Structure Entity and Intersecting Entity elements. The relationships between them are defined by the business rules.

- Principal Entities are identified from the Business Entities in scope
- Intersecting Entities are used to break a many-to-many association between Principal Entities, which form potential business processes
- Structure Entities represent the existence of a potential knowledge base

This is an example of a valid Data Map diagram:

Cluster Reports and Process Maps are deliverables of a valid Data Map diagram analysis.

Perform a Data Map diagram analysis

With the Data Map diagram to be analyzed open and active, either:

- Select the 'Specialize > Add-Ins > Zachman Framework > Do Data-Map Analysis' ribbon option, or
- Right-click on the Data Map diagram in the Browser window, and select the 'Specialize | Zachman Framework | Do Data-Map Analysis' context menu option

The 'Data Map Analysis' dialog displays.
Click on the checkbox against each deliverable required. If you have selected 'Generate Cluster Report', also enter the file pathname under which to save the report.

Click on the Generate button.
Cluster Report

A cluster is a logically related group of processes arranged in a sequence, this being the plan of the order of the execution of processes.

This Cluster Report was generated for the sample Data Map diagram, in .rtf format.

---

Semantic Data Model - Cluster Report
Date Created: 24/10/2014 04:50:14 PM

1. Customer Account Management Cluster
   1. Repayment Plan
      2. Account
         1. Credit
      1. Customer
         2. LoanApplication
         1. Loan
            2. Loan Appl Processing
               3. Customer Credit (Customer Credit Management)
      1. Accounting
      2. Repayment
         3. Customer Account (Customer Account Management)

2. Customer Credit Management Cluster
   1. Credit
      1. Repayment Plan
      2. Account
      1. Accounting
      2. Repayment
         3. Customer Account (Customer Account Management)
   1. Customer
      2. LoanApplication
      1. Loan
         2. Loan Appl Processing
            3. Customer Credit (Customer Credit Management)

The report shows how each cluster is a logical group of processes or tasks forming a major business process.
The number preceding each entity name is the phase number for the entity. Phase 1 against an entity means that the entity forms a potential resource/element that must be procured/framed before proceeding with the business process.
Entities with phase numbers greater than 1 are potential processes, with their sequence of execution set after procuring/framing the phase 1 entities in the cluster.
After successful completion of Data Map analysis, the phase property of each entity in the Data Map diagram is set accordingly.

Acknowledgement
The algorithm for Cluster Report generation is derived from the book *Enterprise Architecture for Integration: Rapid Delivery Methods and Technologies* (Clive Finkelstein; April 2006).
Process Map

A Process Map is the visual model of the Cluster Report; however, the Phase 1 entities in the Cluster Report are not shown. The Process Map groups the identified Business Processes into the stages of the project, arranged as a guide for the project.

This is the Process Map generated for the sample Data Map diagram.
# Business Scorecard Report Template

To aid your strategic management methods, the Zachman Framework provides a report template for creating Business Scorecards.

## Generate a Business Scorecard

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>In the Browser window, click on the Package containing your Business Perspectives and Strategies (an Owner</td>
</tr>
</tbody>
</table>
| 2    | Either:  
  - Press F8, or  
  - Select the 'Publish > Model Reports > Report Builder > Generate Documentation' menu option  
    The 'Generate Documentation' dialog displays. |
| 3    | In the 'Use Template' field, click on the drop-down arrow and select 'Balanced Score Card'. |
| 4    | Click on the Generate button. |
Model Validation

The Zachman Framework registers with Enterprise Architect to receive model validation requests from users.

Configure Model Validation

To configure Enterprise Architect to perform Zachman Framework model validation, select:

- 'Design > Model > Manage > Validate > Configure Validation Rules'

The 'Model Validation Configuration' dialog displays.

To perform validation on Zachman Framework models only, click on the Select None button and then click on the checkbox for 'Zachman Framework (ZF) Rules'. Click on the OK button.

Validate Zachman Framework Model

You can validate, against the Zachman Framework rules:

- An element and any connectors attached to it
- A diagram and all its elements, or
- A Package and all its diagrams and elements

To do this, click on the element, diagram or Package and then select:

- 'Design > Model > Manage > Validate > Validate Current Package'

The 'Model Validation status' dialog displays, showing the progress of the validation.
Validation Messages for Elements

These error messages can be output by the validation of a Zachman Framework element.

Messages

<table>
<thead>
<tr>
<th>Element</th>
<th>Diagram and Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event Node</td>
<td>Event Schedule</td>
</tr>
<tr>
<td></td>
<td>Message: Event Nodes must be used only with Business Cycles</td>
</tr>
<tr>
<td></td>
<td>Meaning: An Event Node has been used with elements other than Business Cycle.</td>
</tr>
<tr>
<td>Event Node</td>
<td>Event Schedule</td>
</tr>
<tr>
<td></td>
<td>Message: Message triggered Event Node must have a message defined</td>
</tr>
<tr>
<td></td>
<td>Meaning: An Event Node with the 'Trigger' Tagged Value set to 'Message' does not have the 'MessageDetail' Tagged Value set.</td>
</tr>
<tr>
<td>Event Node</td>
<td>Event Schedule</td>
</tr>
<tr>
<td></td>
<td>Message: Rule triggered Event Node must have Rule defined</td>
</tr>
<tr>
<td></td>
<td>Meaning: An Event Node with the 'Trigger' Tagged Value set to 'Rule' does not have the 'Rule' Tagged Value set.</td>
</tr>
<tr>
<td>Event Node</td>
<td>Event Schedule</td>
</tr>
<tr>
<td></td>
<td>Message: Error triggered Event Node must have the Error defined</td>
</tr>
<tr>
<td></td>
<td>Meaning: An Event Node with the 'Trigger' Tagged Value set to 'ErrorDetail' does not have the 'Error' Tagged Value set.</td>
</tr>
<tr>
<td>Event Node</td>
<td>Event Schedule</td>
</tr>
<tr>
<td></td>
<td>Message: Multiple triggered Event Node must have a defined list of Triggers</td>
</tr>
<tr>
<td></td>
<td>Meaning: An Event Node with the 'Trigger' Tagged Value set to 'Multiple' does not have the 'Trigger' Tagged Value set.</td>
</tr>
<tr>
<td>Business Cycle</td>
<td>Event Schedule</td>
</tr>
<tr>
<td></td>
<td>Message: Business Cycles must have Event Nodes defined</td>
</tr>
<tr>
<td></td>
<td>Meaning: A Business Cycle element does not have any Event Nodes defined.</td>
</tr>
<tr>
<td>Goal</td>
<td>Business Motivation/ Strategy Map</td>
</tr>
<tr>
<td></td>
<td>Message: Goal not realized</td>
</tr>
<tr>
<td></td>
<td>Meaning: A Goal has no relationship defined with other model artifacts.</td>
</tr>
<tr>
<td>Strategy</td>
<td>Business Motivation/ Strategy Map</td>
</tr>
<tr>
<td></td>
<td>Message: Strategy not realized</td>
</tr>
<tr>
<td></td>
<td>Meaning: A Strategy has no relationship defined with other model artifacts.</td>
</tr>
</tbody>
</table>
Validation Messages for Connectors

These error messages can be output by the validation of a Zachman Framework connector.

**Messages**

<table>
<thead>
<tr>
<th>Connector</th>
<th>Diagram and Message</th>
</tr>
</thead>
</table>
| Association | Data Map  
Message: DataMap Association must have a valid source element  
Meaning: An Association has a source element other than Principal Entity, Structure Entity or Intersecting Entity. |
| Association | Data Map  
Message: DataMap Association must have a valid target element  
Meaning: An Association has a target element other than Principal Entity, Structure Entity or Intersecting Entity. |
| Association | Data Map  
Message: Possibility of an Intersecting entity < name> which might represent a Potential Business Process exists – This is a warning message.  
Meaning: An Association has a many-to-many relationship, informing that the relationship could be normalized. |
| Strategy Link | Strategy Map  
Message: StrategyMap Association must have a valid source element  
Meaning: A Strategy Link has a source element other than Strategy and Goal. |
| Strategy Link | Strategy Map  
Message: StrategyMap Association must have a valid target element  
Meaning: A Strategy Link has a target element other than Strategy and Goal. |
Validation Messages for Diagrams

These error message can be output by the validation of a Zachman Framework diagram.

Messages

<table>
<thead>
<tr>
<th>Diagram</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Map</td>
<td>Entities must have relations in DataMap</td>
</tr>
<tr>
<td></td>
<td>Meaning: In the Data Map diagram there are entities with no relationships defined.</td>
</tr>
</tbody>
</table>
ArchiMate

The MDG Technology for ArchiMate®3.1 is one of the Enterprise Architecture tools integrated with Enterprise Architect.

ArchiMate® is an open-standard enterprise architecture language managed by The Open Group, based on the IEEE 1471 standard. It is "a visual language with a set of default iconography for describing, analyzing, and communicating many concerns of Enterprise Architectures as they change over time." It is used by a variety of modelers and stakeholders; Enterprise Architects will create and maintain models but a range of other stakeholders will view the models and representations of them. These include Solution Architects, Business Analysts, Executives and Business line managers, just to mention a few. The language is used to represent elements from all layers and aspects of an organization, from Drivers and stakeholders to Devices and Networks.

Overview of Topics

The table below lists the main topics that describe the implementation of the ArchiMate language in Enterprise Architect. The architect will learn how to start modeling with ArchiMate and will be introduced to the features available within the tool to create expressive and powerful diagrams and views with ArchiMate. Further topics will describe the creation of Views and Viewpoints and how to create diagrams that contain elements and relationships and will detail all these concepts. Later topics discuss the migration from earlier versions of the language to the latest version and also how to exchange models.

<p>| Welcome | This topic provides an introduction to ArchiMate in Enterprise Architect and discusses its relevance and importance in creating and maintaining Enterprise Architecture models. |
| Getting Started | This topic provides the information needed to immediately start modeling with ArchiMate, including the ArchiMate perspective and the Model Wizard patterns. |
| Using ArchiMate | This topic covers the basic tool features to create Packages, diagrams and elements, including showing how existing and new elements can be added to diagrams. |
| Views and Viewpoints | This topic describes the archimate views and viewpoints giving examples of each viewpoint available for immediate creation from the model wizard pattern tool. |
| ArchiMate Language | This topic describes the ArchiMate Language, including the language definitions, structure and metamodel, and the concepts, divided into elements and relationships. |
| Customising the Language | This topic describes the principles and mechanisms for extending the language to add industry- or organization-specific information in the form of properties, to both elements and connectors. |</p>
<table>
<thead>
<tr>
<th>Topic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exchanging Models</td>
<td>This topic describes the exchange of ArchiMate models created in Enterprise Architect with other tools, enabling models to be both imported and exported using the xml-based exchange format.</td>
</tr>
<tr>
<td>Version Migration</td>
<td>This topic describes how to upgrade models containing elements, relationships and diagrams from one version of ArchiMate to a later version.</td>
</tr>
</tbody>
</table>
Welcome to ArchiMate with Enterprise Architect

Welcome to the ArchiMate language, fully integrated with Enterprise Architect. The tool not only supports all of the language features, including productivity features, but also provides a powerful collaboration platform that will allow architects and others from management down to development and support teams to view and contribute to the architecture and the value it brings to the organization.

About ArchiMate

ArchiMate is a technical standard managed by The Open Group; it is partly inspired by and based on the concepts of the IEEE 1471 standard (Recommended Practice for Architecture Description of Software-Intensive Systems). Work began on the standard as early as 2002 in the Netherlands by a project team from the Telematica Instituut in association with a number of Dutch partners from government, industry and academia. It has evolved significantly and has become a general purpose standard that is used for analyzing, specifying and documenting architectures at any level from strategy to deployment.

Enterprise Architect has supported modeling in ArchiMate for a number of years; the current version of Enterprise Architect supports ArchiMate 3.1 and all its syntax, semantics and viewpoints.

Objectives

"The standard is the specification of the ArchiMate Enterprise Architecture modeling language, a visual language with a set of default iconography for describing, analyzing, and communicating many concerns of Enterprise Architectures as they change over time. The standard provides a set of entities and relationships with their corresponding iconography for the representation of Architecture Descriptions." ArchiMate ® 3.1 Specification p.1

Value

The language has been developed with a recognition of the varied stakeholder groups both technical and non-technical that need to interact with the architecture and has provided a large number of features that support this varied group and their need to communicate and collaborate. The language provides:

- a uniform representation for diagrams that describe Enterprise Architectures,
- views and viewpoint to address stakeholder specific concerns,
- mechanisms to visualize interrelated architectures including different domains,
- syntactic devices for structuring mechanism for architecture domains, layers, and aspects,
- visual devices that allow language elements to be presented with tailored notations for stakeholder groups,
- support for conceptualization by relation abstract and concrete concepts using realization,
- service-orientation to distinguish and relate the Business, Application, and Technology Layers of Enterprise Architectures
Getting Started with ArchiMate

Enterprise Architect supports a wide range of languages and frameworks, which are all available within the tool depending on the edition that you are running. This provides great flexibility as it allows languages to be used in combination; for example, even though most of your architecture might be developed using ArchiMate there could be occasions when you need to create a Mind Map to document a stakeholder workshop or run a simulation of a Business Process to fully understand it. The tool provides a series of Perspectives that let you select a single aspect, language or discipline within the tool, allowing you to focus, but when you need to you can simply switch to an alternative perspective.

ArchiMate Perspective

The ArchiMate perspective allows you to focus on architecture analysis and modeling using the ArchiMate language. To switch to the ArchiMate perspective you need to:

1. Click the Perspective Selector in the top right corner of the application title bar.

2. Select Enterprise Architecture > ArchiMate from the drop down

Enterprise Architect now will allow you to focus on modeling with ArchiMate and will limit the available diagrams and toolbox and language features to the ArchiMate technology. It will also open the Model Wizard, allowing you to kick-start your modeling with a series of pre-built patterns that fully supports the ArchiMate Viewpoint mechanism.

Model Wizard Patterns

The Model Wizard is a productivity tool that allows you to create compliant model content based on the chosen example viewpoint, selected from these viewpoint groups:

- Basic Viewpoints
- Motivation Viewpoints
- Strategy Viewpoints
- Implementation and Migration Viewpoints

The patterns contained in the Model Wizard window are fully documented and provide valuable information that describe the viewpoint and how it can be used.

Basic Viewpoints
Figure: Showing the basic viewpoint with elements from a number of layers.

**Motivation Viewpoints**

Figure: Showing the viewpoints in the Motivation group that relate elements such as Stakeholders and Drivers.

**Strategy Viewpoints**

Figure: Showing the viewpoints in the Strategy group showing elements such as Capabilities and Resources.

**Implementation and Migration Viewpoints**
The pattern documentation provides a detailed description of the pattern (in this case a viewpoint) and an example diagram that shows the user what will be generated, including the model content and details of how to use the pattern. This illustration shows the documentation for one of the most important Strategy Viewpoints - the Capability Map viewpoint, which allows Business Architects to create a structured overview of the capabilities of the enterprise.

**Capability Map Viewpoint**

The Capability Map Viewpoint pattern creates elements and a diagram that allows Capabilities to be visualized in a nested hierarchy. The Capabilities are also nested in a hierarchy in the Project Browser allowing groups of them to be easily moved from one location to another. Color has been used to convey the levels of the hierarchy.

The documentation clearly describes the viewpoint and provides images that not only show the diagrams that will be created, but also show a screenshot of the Browser window, the elements that would be created and the resulting repository structure. The discussion provides tips that will help the business architect work with the pattern, including next steps and how to manipulate the pattern elements. There is also a list of Help topics and a set of tools that can be used when working with the pattern.
Using ArchiMate with Enterprise Architecture

With the ArchiMate perspective selected, as shown in the Getting Started topic, all of the language features such as concepts, diagrams and views and viewpoints will be available to the modeler. Enterprise Architect also provides a way of creating a repository structure using Packages that will act as the containers for the elements and diagrams that you create to describe your enterprise. There is also a wide range of tools that are useful for working with ArchiMate models including diagram filters, legends, notes and tools for navigating and searching, which will be useful as your models get larger.

Our world has fundamentally shifted to architects working together, often in distributed settings, and this is important as Enterprise Architecture models often grow organically with a number of architects contributing to a central model. Enterprise Architect is essentially a collaboration platform that allows architects and other stakeholders and contributors to work together, sharing ideas and using the discussion, review and other collaboration features to ensure that robust and relevant architectures are created.

Adding Diagrams

Diagrams are one of the most powerful ways of communicating with other team members and with stakeholders who have an interest in the enterprise architecture. Diagrams can be created in three different ways:

- An empty diagram can be created and existing elements can be added from the Browser window, or new elements and connectors can be added from the Diagram Toolbox
- A diagram can be created from a user-defined pattern that also contains elements and connectors

In the next section we will also explore another method by which a diagram can be created using the model wizard tool. As a modeler you are likely to use all of these methods at different times, depending on the circumstances and the modeling context. To create a new ArchiMate diagram you can use one of these methods, ensuring that you have chosen the ArchiMate perspective.

![Figure: New Diagram dialog showing the ArchiMate diagram types](image)

Access

<table>
<thead>
<tr>
<th>Ribbon</th>
<th>Design &gt; Diagram &gt; Add</th>
</tr>
</thead>
<tbody>
<tr>
<td>Context Menu</td>
<td>Browser window</td>
</tr>
<tr>
<td>Keyboard Shortcuts</td>
<td>&lt;Ctrl&gt; + &lt;Insert&gt;</td>
</tr>
</tbody>
</table>
Creating a Package Structure

There are two approaches to creating a suitable Package structure within Enterprise Architect, both of which mimic the enterprise architecture methods.

- An initial well developed Package structure that is changed very little through the course of model development for an initiative
- A skeleton model that contains the main Packages and is augmented as new needs are understood, and changed significantly during an initiative

Either of these methods can be used, or a team could consider a hybrid approach; either way new Packages need to be created in Enterprise Architect that will act as the containers for new elements and diagrams.

Figure: Showing the Package structure in the 'Project' tab of the Browser window.

Adding Views and Viewpoints

The ArchiMate language defines a series of example viewpoints designed to provide representations that are meaningful and relevant to a variety of stakeholders. These viewpoints are made available in Enterprise Architect through the Model Wizard patterns, which provide a way of creating both repository content and diagrams that show how the elements are connected by relationships. Thus new diagrams can be created.

- A diagram and its elements and connectors can be added using the Model Wizard pattern feature

Figure: Showing the Model Wizard window and the Stakeholder Viewpoint in the Motivation group
Adding Elements and Relationships

Elements can be added to the model directly without the need for a diagram to be created, but it is far more common for a diagram to be the device that is used to add both elements and connectors to the model. Diagrams can be built up with a combination of:

- Existing elements dragged from the Browser
- New elements (or Relationships) dragged from the Diagram Toolbox pages

Adding Elements from the Browser

This diagram shows how elements can be added from the Browser window by dragging and dropping them onto the current open diagram canvas.

![Figure: Showing an existing element being dragged from the Browser window](image)

Adding Elements from the Toolbox

This diagram shows how elements can be added from the Toolbox pages by dragging and dropping elements (or relationships) onto the current open diagram canvas.

![Figure: Showing an existing element being dragged from a Toolbox page](image)

Changing Elements and Relationships

Any element or relationship can be changed including its name and properties. When a change is made to an element in any location, for example in a diagram this change will be reflected in any other diagrams (view) that contains the element or relationship.
Architecture Views and Viewpoints

The Architecture Views and Viewpoints are a powerful mechanism to provide stakeholders with relevant and meaningful visualization of the Enterprise Architecture that has been created in the repository. Every stakeholder - or, more commonly, a stakeholder group - will typically have different concerns and interests and so the View and Viewpoint mechanism provides a robust way of ensuring that they get value from the work that has been done to articulate the Enterprise Architecture. Essentially, a View is what you see when 'looking' from a particular standpoint (Viewpoint).
Overview of Views and Viewpoints

The tool has built-in support in a number of different places for the Viewpoint mechanism. Two of the most important places are the:

- Model Wizard Views and Viewpoints for ArchiMate
- Diagram Views available from the Diagram window

Creating Views Using the Toolbox

Enterprise Architect also supports creating Views from a blank canvas using the Diagram Viewpoint mechanism, which allows you to select an example Viewpoint as the basis for a new diagram. Choosing any of the available Viewpoints results in a new diagram being created, and a Toolbox being displayed in which the Viewpoint page only displays the available elements for that Viewpoint.

![Figure: Shows the views available rom the new diagram dialog.](image)

In the next diagram we see two Toolbox pages that are applicable for the Application Usage Viewpoint. The first page contains a subset of the Application Layer elements used for the Viewpoint and the second page contains a subset of the Business Layer elements.
Elements can be dragged and dropped from the Toolbox pages to the open diagram, which will create a new element and add it to the open diagram. Applicable elements can also be dragged onto the diagram from the Browser window, which allows existing elements that have been created or used in other diagrams to be added to the current diagram. For example, an Application Service and a Business Process might have been created at an earlier time and these could be added to a diagram that contains a new Application Component added to an Application Usage Viewpoint.

In this diagram we see that a modeler outside the Enterprise Architect group has created a UML Realization relationship from a UML Component in the Implementation model to an ArchiMate Component, thus linking the two different disciplines without breaking any ArchiMate language syntax rules.
Creating Views Using the Wizard

Enterprise Architect has full support for the views and viewpoints and has a powerful Model Wizard patterns where each viewpoint has been conveniently created and can be injected into the repository at any location. This image shows the icon to select in the Browser window toolbar to open the Model Wizard.

Tracking Elements between Views

Enterprise has a number of windows that allow elements to be tracked between views and more broadly between any diagrams that contain the elements. This can be achieved by using the element's context menu and selecting 'Find in all Diagrams'. When selected, and when an element does appear in multiple views, a window will provide a list of all the views that contain the element.
Basic Viewpoints

The basic Viewpoints are a reference or starting point for architectural descriptions and for creating meaningful and useful models. They act as a catalyst for an architect, allowing that person to get some modeled content down, thus removing the ‘canvas-fright’ syndrome that tends to paralyze progress. They are useful in supporting peer reviews, fostering organizational standards and helping novice modelers or newcomers to a domain. The ArchiMate specification has included all the Viewpoints as examples and it is well to remember that they should be tailored to suit the stakeholders who will ultimately need to digest the material. The basic Viewpoints are primarily intended for the architecture community. Some Viewpoints are limited to a single layer while others can contain elements from multiple layers.

Table of Basic Viewpoints

The following table list the example Basic Viewpoints.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organization</td>
<td>Structure of the enterprise in terms of roles, departments, etc.</td>
</tr>
<tr>
<td>Application Structure</td>
<td>Shows the structure of a typical application in terms of its constituents.</td>
</tr>
<tr>
<td>Information Structure</td>
<td>Shows the structure of the information used in the enterprise.</td>
</tr>
<tr>
<td>Technology</td>
<td>Infrastructure and platforms underlying the enterprise’s information systems in terms of networks, devices, and system software.</td>
</tr>
<tr>
<td>Layered</td>
<td>Provides overview of architecture(s).</td>
</tr>
<tr>
<td>Physical</td>
<td>Physical environment and how this relates to IT infrastructure.</td>
</tr>
<tr>
<td>Product</td>
<td>Shows the contents of products.</td>
</tr>
<tr>
<td>Application Usage</td>
<td>Relates applications to their use in, for example, business processes.</td>
</tr>
<tr>
<td>Technology Usage</td>
<td>Shows how technology is used by applications.</td>
</tr>
<tr>
<td>Business Process</td>
<td>Shows the relationships between various business processes.</td>
</tr>
<tr>
<td>Cooperation</td>
<td>Shows application components and their mutual relationships.</td>
</tr>
<tr>
<td>Service Realization</td>
<td>Shows how services are realized by the requisite behavior.</td>
</tr>
<tr>
<td>Implementation and</td>
<td>Shows how applications are mapped onto the underlying technology.</td>
</tr>
<tr>
<td>Deployment</td>
<td></td>
</tr>
</tbody>
</table>
Organization Viewpoint

The organization viewpoint focuses on the (internal) organization of a company, department, network of companies, or of another organizational entity. It is possible to present models in this viewpoint as nested block diagrams, but also in a more traditional way, such as organizational charts. The organization viewpoint is very useful in identifying competencies, authority, and responsibilities in an organization.

The following image shows a diagram created from the Viewpoint Pattern that is a built-in part of the Model Pattern facility.

![Diagram of the Organization viewpoint](image)

Figure: Showing the Organization viewpoint
Application Structure Viewpoint

The application structure viewpoint shows the structure of one or more applications or components. This viewpoint is useful in designing or understanding the main structure of applications or components and the associated data; e.g., to break down the structure of the system under construction, or to identify legacy application components that are suitable for migration/integration.

The following image shows a diagram created from the Viewpoint Pattern that is a built-in part of the Model Pattern facility.

Figure: Showing the Application Structure viewpoint
Information Structure Viewpoint

The *Information Structure Viewpoint* pattern creates elements that show the structure of the information used in the enterprise or in a specific business process or application, in terms of data types or information elements. It will assist in visualizing the information from the business level through the application level down to the infrastructure elements that implement databases and other persistent stores.

![Diagram](image)

Figure: Showing the *Information Structure Viewpoint* viewpoint
Technology Viewpoint

The Technology Viewpoint pattern creates elements and a diagram that describes the software and hardware technology elements supporting the Application Layer, such as physical devices, networks, or system software such as middleware operating systems, databases and other containers.

Scope: Technology layer - Multiple aspects

The following image shows a diagram created from the Viewpoint Pattern that is a built-in part of the Model Pattern facility.

Figure: Showing the Technology Viewpoint
Layered Viewpoint

The *Layered Viewpoint* pattern creates a number of elements and diagrams that allow the visualization of multiple layers of an Enterprise Architecture in a single diagram. The partitioning which uses the Grouping element allows the representation of elements such as Business Processes in dedicated layers and elements such as Application Services in services layers. Any number of layers can be included but the diagram is most expressive when dedicated and service layers are interleaved.

The following image shows a diagram created from the Viewpoint Pattern that is a built-in part of the Model Pattern facility.
Figure: Showing the *Layered Viewpoint*
Physical Viewpoint

The *Physical Viewpoint* pattern creates elements and diagrams that contains equipment (one or more physical machines, tools, or instruments) that can create, use, store, move, or transform materials. It also describes how the equipment is connected via the distribution network and allows the visualization of other active elements that are assigned to the equipment.

![Diagram showing physical viewpoint](image)

Figure: Showing the *Physical Viewpoint*
Product Viewpoint

The *Product Viewpoint* pattern creates elements and a diagram that describe the value that the products offer to external parties such as customers or other stakeholders. It allows them to visualize one or more products' composition in terms of their constituent business, application, or technology services and any number of contracts or other agreements. The channels (interfaces) through which this product is offered, and the events associated with the product can also be represented in this viewpoint.

![Physical Viewpoint](image)

Figure: Showing the *Physical Viewpoint*
Application Usage Viewpoint

The Application Usage Viewpoint pattern creates elements and a diagram that describes how application services and the applications that realize them are used to support any number of business processes. It can also show the relationship between those applications that implement the services.

Figure: Showing the Application Usage Viewpoint
Technology Usage Viewpoint

The Technology Usage Viewpoint pattern creates elements that show how applications are supported by the software and hardware technology: the technology services are delivered by the devices; system software and networks are provided to the applications. This viewpoint plays an important role in the analysis of performance and scalability, since it relates the physical infrastructure to the logical world of applications.

Figure: Showing the Technology Usage Viewpoint
Business Process Cooperation Viewpoint

The Business Process Cooperation Viewpoint pattern creates elements and a diagram that describe the business processes showing how they relate to each other and also with their environment. This includes relationships with Business Services and Business Objects and the Roles and Actors that perform the processes or who are affected by them.
Application Cooperation Viewpoint

The Application Cooperation Viewpoint pattern creates elements a diagram that describe the relationships between applications components and their locations, the services they provide or utilize and the information that flows between them.

Figure: Showing the Application Cooperation Viewpoint
Service Realization Viewpoint

The Service Realization Viewpoint pattern creates elements that show how one or more business services are realized by the underlying processes (and sometimes by application components). Thus, it forms the bridge between the business products viewpoint and the business process view. It provides a “view from the outside” on one or more business processes.

Figure: Showing the Service Realization Viewpoint
Implementation and Deployment Viewpoint

The *Implementation and Deployment Viewpoint* pattern creates elements and a diagram that relate programs and projects to the parts of the architecture that they implement. This view allows modeling of the scope of programs, projects and project activities in terms of the plateaus that are realized or the individual architecture elements that are affected. In addition, the way the elements are affected can be indicated by annotating the relationships.

![Figure: Showing the Implementation and Deployment Viewpoint](image)

(c) Sparx Systems 2020
Page 252 of 328
Created with Enterprise Architect
## Motivation Viewpoints

### Table of Motivation Viewpoints

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stakeholder</td>
<td>Focuses on modeling the stakeholders, drivers, the assessments of these drivers, and the initial goals to address these drivers and assessments.</td>
</tr>
<tr>
<td>Goal Realization</td>
<td>Focuses on modeling and analyzing the influence relationships between goals (and requirements).</td>
</tr>
<tr>
<td>Requirements Realization</td>
<td>Focuses on modeling the realization of requirements and constraints by means of core elements, such as actors, services, processes, application components, etc.</td>
</tr>
<tr>
<td>Motivation</td>
<td>Covers the entire motivational aspect and allows use of all motivational elements.</td>
</tr>
</tbody>
</table>
Stakeholder Viewpoint

The Stakeholder Viewpoint pattern creates stakeholders, the internal and external drivers for change, and the assessments (in terms of strengths, weaknesses, opportunities, and threats) of these drivers. Also, the links to the initial (high-level) goals that address these concerns and assessments can be described. These goals form the basis for the requirements engineering process, including goal refinement, contribution and conflict analysis, and the derivation of requirements that realize the goals.

Figure: Showing the Stakeholder Viewpoint
Goal Realization Viewpoint

The *Goal Realization Viewpoint* pattern creates elements and a diagram that models the relationships between goals including the decomposition to sub-goals. The goals are realized by an Outcome which is realized by a Principle which behaves like a more abstract and broader requirement. Finally the Principle is realized by a Requirement indicating specific properties that the system must exhibit.

*Figure: Showing the Goal Realization Viewpoint*
Requirements Realization Viewpoint

The Requirements Realization Viewpoint pattern creates elements and a diagram that model the realization of Goals into Requirements and Constraints and then how these Requirements are realized by core elements such as Business and Application Services. Color has been introduced to add appeal to the diagram and to distinguish the element types.

Figure: Showing the Requirements Realization Viewpoint
Motivation Viewpoint

The *Motivation Viewpoint* pattern creates elements and a diagram that completely covers the motivational aspect from a given stakeholder’s perspective defining a Driver, an Assessment, a number of Goals and the Principle that is applied and the Requirements and Constrains that are needed to qualify the Principle.

Figure: Showing the Requirements Realization Viewpoint
## Strategy Viewpoints

### Table of Strategy Viewpoints

<table>
<thead>
<tr>
<th>Strategy</th>
<th>Provides a high-level strategic overview of the strategies of the enterprise, its capabilities, value streams, and resources, and the envisaged outcomes.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Capability Map</td>
<td>Provides an overview of the capabilities of the enterprise.</td>
</tr>
<tr>
<td>Value Stream</td>
<td>Shows an overview of value-creating steps in the enterprise and the capabilities that support these.</td>
</tr>
<tr>
<td>Outcome Realization</td>
<td>Describes how high-level, business-oriented results are produced by the capabilities and resources of the enterprise.</td>
</tr>
<tr>
<td>Resource Map</td>
<td>Provides a structured overview of the resources of the enterprise.</td>
</tr>
</tbody>
</table>
Strategy Viewpoint

The *Strategy Viewpoint* pattern creates elements and a diagram that models the strategic intent of an organization by articulating a Course of Action and the Capabilities and Resources needed to achieve it providing a modeled Outcome.

![Strategy Viewpoint Diagram](image)

Figure: Showing the *Strategy Viewpoint*
Capability Map Viewpoint

The Capability Map Viewpoint pattern creates elements and a diagram that allows Capabilities to be visualized in a nested hierarchy. The Capabilities are also nested in a hierarchy in the Project Browser allowing groups of them to be easily moved from one location to another. Color has been used to convey the levels of the hierarchy.

Figure: Showing the Capability Map Viewpoint
Value Stream Viewpoint

The value stream viewpoint allows the Business Architect to create a structured overview of a value stream, the capabilities supporting the stages in that value stream, the value created, and the stakeholders involved.

The following image shows a diagram created from the Viewpoint Pattern that is a built-in part of the Model Pattern facility.

Figure: Showing the Value Stream Viewpoint
Outcome Realization Viewpoint

The *Outcome Realization Viewpoint* pattern creates elements and a diagram that model how core elements deliver the high level business value. The diagram is useful to show how the strategic level business elements such as Value and Outcomes are realized by the underlying elements that deliver that value such as Capabilities, Services and Components.

Figure: Showing the *Outcome Realization Viewpoint*
Resource Map Viewpoint

The Resource Map Viewpoint pattern creates a number of Resource elements nested into three layers. It permits a Business Architect or other stakeholder to create a structured overview of the resources available to an enterprise. The map commonly shows two or three levels of resources across an entire enterprise.

Figure: Showing the Resource Map Viewpoint
Implementation and Migration Viewpoints

The Implementation and Migration Viewpoints are designed to allow the modeler to create relevant and meaningful views of the implementation and migration aspects of the architectures including such things as change and transitions.

Table of Implementation and Migration

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Project</td>
<td>Primarily used to model the management of architecture change.</td>
</tr>
<tr>
<td>Migration</td>
<td>Used to model the transition from an existing architecture to a target architecture.</td>
</tr>
<tr>
<td>Implementation and Migration</td>
<td>Used to model the relationships between the programs and projects and the parts of</td>
</tr>
<tr>
<td></td>
<td>the architecture that they implement.</td>
</tr>
</tbody>
</table>
Project Viewpoint

The Project Viewpoint pattern creates elements and diagrams that contain elements that model the management of architecture change. This includes the transition from a baseline to a target enterprise architecture is complex and can be constrained by Portfolio Management, Project Management and a number of other disciplines.

Figure: Showing the Project Viewpoint
Migration Viewpoint

The Migration Viewpoint pattern creates elements and a diagram that model the transition from a baseline to a target enterprise architecture. The Plateau represents a relatively stable state of the architecture that exists during a limited period of time, whereas the Gap represents a statement of the difference between the two states.

![Diagram showing the Migration Viewpoint](image-url)

Figure: Showing the Migration Viewpoint
Implementation and Migration Viewpoint

The **Implementation and Migration Viewpoint** pattern creates elements and a diagram that model relate programs and projects to the parts of the architecture that they implement. This view allows modeling of the scope of programs, projects, project activities in terms of the plateaus that are realized or the individual architecture elements that are affected.

![Diagram showing the Implementation and Migration Viewpoint](c) Sparx Systems 2020

Figure: Showing the *Implementation and Migration Viewpoint*
The ArchiMate Language

Enterprise Architect provides a highly compliant implementation of the ArchiMate standard, allowing Enterprise Architects and others to create rigorous and expressive models of enterprise architecture concerns, from the level of stakeholders and their interests right down to the virtual or physical devices that ultimately provide the computational power to run the enterprise and deliver valued services and outcomes to customers.

The language is deliberately pragmatic and has as a design consideration the need for the syntax and symbols to be compact and succinct, but at the same time provide the expressive power to capture and communicate structural and behavioral aspects of the model. The Enterprise Architect platform, with WebEA and Prolaborate, allows teams to create visualizations curated for any stakeholder group, from senior executives down to implementation staff.

Enterprise Architect implements a large number of other standards, and models produced by other teams can be connected to the Enterprise Architecture models described with ArchiMate, thus creating a tapestry of interwoven threads. For example, ArchiMate deliberately takes a birds-eye view of process models, leaving out much of the details and concentrating on conveying the value and how it relates to Business Functions and Services, Roles and Actors. These models can be connected to more detailed models described the use of Business Process Model and Notation (BPMN). The same is true with respect to application models that, in ArchiMate, are quite high level and intentionally leave out implementation details such as protocols, ports and message flow. These latter details can be modeled using the Unified Modeling Language (UML) and again connected to the ArchiMate Application Components to create expressive models allowing drill-down and drill-up from any modeling context.

Figure: Showing the relationship between ArchiMate and other modeling languages.

Enterprise Architect implements a large number of other standards, and models produced by other teams can be connected to the Enterprise Architecture models described with ArchiMate, thus creating a tapestry of interwoven threads. For example, ArchiMate deliberately takes a birds-eye view of process models, leaving out much of the details and concentrating on conveying the value and how it relates to Business Functions and Services, Roles and Actors. These models can be connected to more detailed models described the use of Business Process Model and Notation (BPMN). The same is true with respect to application models that, in ArchiMate, are quite high level and intentionally leave out implementation details such as protocols, ports and message flow. These latter details can be modeled using the Unified Modeling Language (UML) and again connected to the ArchiMate Application Components to create expressive models allowing drill-down and drill-up from any modeling context.

Figure: The relationship created between an ArchiMate application component and a UML component
Language Definitions

Enterprise Architect fully implements the ArchiMate language and supports all the language concepts and definitions, allowing Business, Application, Technology and Security Architects and others - including Geospatial Architects - to create highly expressive and compliant language models, views and other powerful visualizations of the enterprises under discussion.

ArchiMate Core Framework

The reference structure used to classify elements of the ArchiMate core language is implemented in Enterprise Architecture in its core internal metamodel. It consists of three layers and three aspects (including the physical) and the extended framework contains a number of extra layers and aspects. as extensions. These core and extended language elements are visible in the tool in the form of the Diagram list and tool boxes corresponding to the layers and in the element representations with their various forms including the icon format. The following diagram has been created in Enterprise Architect to demonstrate the structure of the language.

![Diagram showing the layers and aspects of the core framework](image)

ArchiMate Core Language

The central part of the ArchiMate language that defines the concepts to model Enterprise Architectures has been built into the ArchiMate profile and meta model defined within the MDG Technology inside the tool. The user can access all of the defined ArchiMate language concepts and features, including elements, relationships, relationship connectors and Viewpoints from a series of user-friendly tool features.

Architecture View

A representation of a system from the perspective of a related set of concerns typically relating to a single stakeholder or stakeholder group. Enterprise Architect supports this concept with a range of different product tools and mechanisms. The most important ones are:

- **Diagrams** - any number of diagrams can be created that can display particular elements, properties with selected themes and presentation styles and also restricted using filters.
- **Matrices** - elements and their relationships can be displayed in a matrix showing which elements are connected by relationships

- **Textual** - there are a number of tools - including list views, specification views and documents - that allow elements and relationships to be presented in textual form, creating compelling narratives

There is also a Model Wizard pattern tool that allows any of the ArchiMate example Viewpoints to be created. This illustration shows the mechanism to switch between alternative views of the same underlying model content.

![Switch View](image)

Figure: Showing the tool options to switch views

**Architecture Viewpoint**

A specification of the conventions for a particular kind of architecture view. Enterprise Architect provides a powerful productivity and compliance tool called Model Wizard patterns that allows all the example ArchiMate viewpoints to be created from well defined and described patterns built into the ArchiMate perspective. This diagram shows the result of using the Wizard to create a Technology Usage Viewpoint.

![Diagram](image)

Figure: Showing the Technology Usage Viewpoint

**Aspect**
The tool supports the concept of Aspects, which fundamentally divide the corpus of elements into syntactic groups based on grammatical roles. The Aspects allow the modeler and the viewer to understand the model and diagrammatic constructs in terms of the role they play. The Aspects are Active Structure, Behavior, Passive Structure and Motivation. These are visible throughout the tool's implementation of ArchiMate in the Diagram Toolbox, which make new elements available to be added to the model and diagrams.

![Diagram Toolbox with Aspects]

Figure: Showing the elements and relationships grouped by aspect e.g., *Active Structure*

**Attribute**

Enterprise Architect supports a number of ways of adding properties to elements and relationships. Elements can have two different types of property:

- **Built-in Properties:** *e.g.* Name, Notes, Alias, Author
- **Custom Properties:** *e.g.* Business Owner, Technical Owner, End-of-Life Date
It is common for an Enterprise Architecture function or practice to add organization-specific properties that help in the analysis of business, application and technology architectures. These can be added as Tagged Values either directly to an element, or more robustly using the profile system that allows the creation and augmentation of elements as a new type that results in a fully compliant ArchiMate element with the additional properties.

**Composite Element**

The tool supports the creation of the two Composite Elements namely: Grouping and Location which allow aggregation relationships to other elements typically from multiple aspects or layers of the language.

![Diagram](image.png)

**Core Element**

Enterprise Architect supports all the core elements defined in the ArchiMate specification, in all aspects and all layers including the extensions.
Element

The tool supports the creation and modification of all the ArchiMate elements across all layers and aspects, and provides mechanisms to place these elements onto diagrams that comply with the example viewpoints, including allowing them to participate in relationships with other elements (and, in some cases, other relationships) in accordance with the defined combinational rules. Elements, along with Relationships and Relationship Connectors, are the most primitive type of concept in the ArchiMate metamodel and are used to define and describe the constituent parts of Enterprise Architectures and their unique set of characteristics.

Figure: Showing the Product Viewpoint

In this diagram we see elements from a number of different layers, using a number of different relationships, including nesting as an alternative. Some of the elements are presented using their rectangular style and others (e.g. Application Services) are presented using their icon style. The style can be toggled in Enterprise Architect to suit the audience and stakeholders.

Layer

Enterprise Architect is a powerful platform for architectural description and visualization and supports a variety of domain architects which are represented in the language and the tool by the concept of a layers. These domain specific models can be articulated with each other without ambiguity and are visible in the tool in the form of diagrams and their accompanying toolboxes.
Model

The tool provides a powerful repository based solution which includes the ability to create any number of models either in a single repository for the purposes of collaboration and model sharing or in separate and isolated repositories for the purpose of isolation or the separation of concerns. Regardless of how the models are organized architects and other stakeholders can access the models via the Internet in desktop and browser-based tools.

Relationship

Enterprise Architect supports the creation and modification of all the ArchiMate relationships and relationship connectors across all layers and aspects, and provides mechanisms to place these relationships onto diagrams, thus connecting elements (and in some cases other relationships) in compliance with the specification. The tool has a productivity tool called the Quick Linker, which allows relationships to be created by dragging from the corner of a source element to a target element. The Quick Linker restricts the available relationships for a source-target pair to the set of permissible relationships based on the specification. These are defined in the Sparx Systems ArchiMate metamodel that underpins the implementation and ArchiMate Technology.
Figure: Example Quick Linker menu showing the restricted set of relationships for the elements
Language Structure

When you create architectures in Enterprise Architect you are fundamentally creating ArchiMate Models that are made up of a collection of the following concepts:

- Elements
- Relationships
- Relationship Connectors

These are indicated in this metamodel diagram.

![Metamodel Diagram]

Elements of one of four types can be created and will appear in the Browser window:

- Structure Elements
- Behavior Elements
- Motivation Elements
- Composite Elements
Generic Metamodel

The main hierarchy of behavior and structure elements of the ArchiMate language is shown in this diagram. All the elements in this model have italic names, indicating that they are abstract and do not themselves participate in models. They are represented as independent of the layers of the framework as they can apply across all layers. The most fundamental division of the language elements is into Behavior and Structure elements.

Figure: Showing the fundamental hierarchy of the ArchiMate elements.

Elements such as an Application Component - which is a type of Active Structure element - and Application Process - which is a type of behavior element - are added to models of architecture.
Elements

Enterprise Architect provides a rich and powerful user interface for working with elements. It allows all the Archimate elements across all aspects and layers to be created, managed and visualized. This includes the following types of elements corresponding to the Archimate toolboxes and their pages.

1. Motivation (Aspect) Elements
2. Strategy (Layer) Elements
3. Business (Layer) Elements
4. Application (Layer) Elements
5. Technology (Layer) Elements
6. Implementation and Migration (Layer) Elements
7. Composite Elements
Changing Element Presentation

An Enterprise Architect will typically need to provide different visualizations for different groups of stakeholders, and a diagram that is suitable for one group might not be appropriate for another. Enterprise Architect provides a wide range of ways for working with elements and diagrams, including how the elements are presented and visualized in different diagrams and different windows within the product.

Enterprise Architect has great flexibility and, because the tool is implemented using a robust relational database back-end, when changes are made to a single element on a diagram or any other view the changes can be automatically applied to all places in diagrams or other windows where the element appears. This applies to element properties, name changes and other fundamental aspects of the element. When the appearance (such as color and style) of an element is changed a modeler has a choice of:

- Changing the diagram Object style (just applies to the current diagram)
- Changing the default Element Appearance (applies to the current diagram AND on all other diagrams)

Changing Size, Proportion and Color

Any of the standard ArchiMate elements can be changed to different sizes, proportions and colors without affecting the meaning or syntax of the diagrams. Enterprise Architect provides great flexibility for changing both the elements and the diagram styles and themes. This illustration shows two different diagrams where the same elements have been rendered in different sizes, proportions and colors.

Changes on a Single Diagram

Changes can be made to the style of an object (element) on a single diagram, once the diagram is opened and the object is selected there are a number of places where the element style can be changed. These changes will only apply to selected element in the diagram and other diagrams that contain the element will not be changed in any way. In the next section we will see how if required the changes can be made update all other diagrams.

The options to change the style are available from a number of locations including the ribbons and the element quick style icon which is displayed when the element is selected in the diagram. The following screen shot shows the options available from the diagram.
Changes on All diagrams

Changes can be made to the style of an object (element) on all diagrams, once the diagram is opened and the object is selected there are a number of places where the element style can be changed. These changes will apply to selected element in the diagram and all other diagrams that contain the element will also be changed in any way. In the previous section we saw how if required the changes can be applied to an element on a single diagram.

The options to change the style are available from a number of locations including the ribbons and the element quick style icon which is displayed when the element is selected in the diagram. The following screen shot shows the options available from the diagram.

Element Usage

Elements from the repository can participate in any number of views (including the null case where they do not appear in any views). The inclusion of a variety of elements on a given diagram and the inclusion of a given element on multiple diagrams is what gives the language its expressive power. Enterprise Architect supports all these language mechanisms and also allows you to locate the different diagrams that contain a particular element.
Selecting this option provides a list of all the diagrams that contain the element, allowing the modeler to hyperlink and open any of the listed diagrams. This provides a convenient way for a modeler to understand the basis for the element's usage as the diagram's type and name are included in the table of usages.

### Rectangular and Icon Views

The ArchiMate language provides the flexibility of displaying a number of elements either as rectangles with a small icon in the top right hand corner or in Icon view where the whole element takes on the Icon shape. As an Icon, wherever possible the name of the element is placed inside the shape, but with some Icons there is no room and the name is positioned underneath the element. Not all elements have an Icon representation and in these cases the icon view cannot be selected.
The tool makes it easy to toggle between the two views, but care must be taken as the geometry of the element will change as the icons are typically smaller than the rectangular representation.

To show an element using rectangle notation, either:

- Click on the element in the diagram and on the rectangle icon that displays on the right of the element, or
- Right-click on the element on the diagram and select the 'Advanced | Use Rectangle Notation' context menu option

This displays the element as a rectangle, by default displaying an ArchiMate icon in the top right corner of the element (where the language defines an icon).

The setting only applies to the selected element, and can be toggled on and off either by:

- Clicking on the icon again, or
- Deselecting the context menu option, or
- Selecting the reciprocal context menu option such as 'Use Circle Notation' or 'Use Actor Notation'
# Motivation Elements

## Motivation Elements

A driver represents an external or internal condition that motivates an organization to define its goals and implement the changes necessary to achieve them.

<table>
<thead>
<tr>
<th>Element</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Stakeholder</strong></td>
<td>A stakeholder represents the role of an individual, team, or organization (or classes thereof) that represents their interests in the effects of the architecture. The image demonstrates both standard and alternative element notation.</td>
</tr>
<tr>
<td><strong>Driver</strong></td>
<td>A driver represents an external or internal condition that motivates an organization to define its goals and implement the changes necessary to achieve them.</td>
</tr>
<tr>
<td><strong>Assessment</strong></td>
<td>An assessment represents the result of an analysis of the state of affairs of the enterprise with respect to some driver.</td>
</tr>
<tr>
<td><strong>Goal</strong></td>
<td>A goal represents a high-level statement of intent, direction, or desired end state for an organization and its stakeholders.</td>
</tr>
<tr>
<td><strong>Outcome</strong></td>
<td>An outcome represents an end result.</td>
</tr>
<tr>
<td><strong>Principle</strong></td>
<td>A principle represents a statement of intent defining a general property that applies to any system in a certain context in the architecture.</td>
</tr>
<tr>
<td><strong>Requirement</strong></td>
<td>A requirement represents a statement of need defining a property that applies to a specific system as described by the architecture.</td>
</tr>
<tr>
<td>Constraint</td>
<td><em>A constraint represents a factor that limits the realization of goals.</em></td>
</tr>
<tr>
<td>------------</td>
<td>---------------------------------------------------------------------</td>
</tr>
<tr>
<td>Meaning</td>
<td><em>Meaning represents the knowledge or expertise present in, or the interpretation given to, a concept in a particular context.</em></td>
</tr>
<tr>
<td>Value</td>
<td><em>Value represents the relative worth, utility, or importance of a concept.</em></td>
</tr>
</tbody>
</table>
Motivation Example Diagrams

Using Enterprise Architect a modeler can create any number of diagrams using the ArchiMate diagrams and Toolbox palettes. The motivation elements can be connected together to create expressive narratives that describe an Enterprise Architecture and the motivation for making business or technology changes articulated in other parts of the architecture. These diagrams have been taken from the Open Group's ArchiMate Specification.

Goal, Outcome, Principle, Requirement, and Constraint

This diagram is concerned with the relationship between Goals Principles, Requirements, and Constraints. The goal “Improve Profitability of Service Offering” is realized by the outcome “Increased Profit by 10% in Next Fiscal Year” and “Reduced Cost of Customer Acquisition by 25%”. We see the use of the influence relationship between outcomes with the indicator of (+) or (-) effect. The outcomes are in turn influenced by a number of principles and the principles are realized by a number of requirements. We also see the use of the OR Junction in the bottom left of the diagram.

Figure: Example diagram from the ArchiMate Specification showing
Stakeholder, Driver, and Assessment

This diagram is concerned with Drivers and the Stakeholders that are concerned about them, which in this example are at the executive level. We see that the Driver entitle Profitability has been decomposed into two other drivers namely: Revenue and Costs. Assessments have been associated with the Drivers which paint a clear picture of the state of the organization. Influence relationship (which can be drawn between any two motivation elements) show how the assessments are related to each other. The (+) indicating a positive influence and (-) indicating a negative influence. This is example 18 in the Motivation Chapter of the specification.

![Diagram of Stakeholder, Driver, and Assessment]

Figure: Example diagram from the ArchiMate Specification showing

Meaning and Value

This diagram is concerned with stakeholders and the value they derive from the system. The notification message has been specialized into three different types. Meanings have been associated with the messages:

- A “Confirmation Of Receipt” message” has the meaning “Claim Has Been Received”
- A “Review Complete” message has the meaning “Claim Review Complete”
- A “Payment Complete” message” has the meaning “Claim Has Been Paid”
Figure: Example diagram from the ArchiMate Specification showing how Meaning and Value are used.
Strategy Layer Elements

Table of Strategy Elements

<table>
<thead>
<tr>
<th>Capability</th>
<th>A capability represents an ability that an active structure element, such as an organization, person, or system, possesses.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value Stream</td>
<td>A value stream represents a sequence of activities that create an overall result for a customer, stakeholder, or end user.</td>
</tr>
<tr>
<td>Course of Action</td>
<td>A course of action represents an approach or plan for configuring some capabilities and resources of the enterprise, undertaken to achieve a goal.</td>
</tr>
<tr>
<td>Resource</td>
<td>A resource represents an asset owned or controlled by an individual or organization.</td>
</tr>
</tbody>
</table>
Strategy Layer Example Diagrams

Capability, Resource, and Course of Action

This diagram is concerned with an overarching Goal to Increase Profit that is made up of two other Goals (these may be considered to be Objectives) namely Decrease Costs and Increase Revenue. Then two Outcomes are modeled that influence the delivery of the Goals. The business architect has modeled two Course of Action elements that influence these Outcomes. The Course of Action elements are Realized by two Capabilities one of which has two Resources Assigned at Headquarters.

Figure: Example diagram from the ArchiMate Specification showing Capabilities, Courses of Actions and Resources.
# Business Layer Elements

## Table of Business Layer Elements

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Business Actor</strong></td>
<td>A <em>business actor</em> represents a business entity that is capable of performing behavior.</td>
</tr>
<tr>
<td><img src="image" alt="Business Actor" /></td>
<td></td>
</tr>
<tr>
<td><strong>Business Role</strong></td>
<td>A <em>business role</em> represents the responsibility for performing specific behavior, to which an actor can be assigned, or the part an actor plays in a particular action or event.</td>
</tr>
<tr>
<td><img src="image" alt="Business Role" /></td>
<td></td>
</tr>
<tr>
<td><strong>Business Collaboration</strong></td>
<td>A <em>business collaboration</em> represents an aggregate of two or more business internal active structure elements that work together to perform collective behavior.</td>
</tr>
<tr>
<td><img src="image" alt="Business Collaboration" /></td>
<td></td>
</tr>
<tr>
<td><strong>Business Interface</strong></td>
<td>A <em>business interface</em> represents a point of access where a business service is made available to the environment.</td>
</tr>
<tr>
<td><img src="image" alt="Business Interface" /></td>
<td></td>
</tr>
<tr>
<td><strong>Business Process</strong></td>
<td>A <em>business process</em> represents a sequence of business behaviors that achieves a specific result such as a defined set of products or business services.</td>
</tr>
<tr>
<td><img src="image" alt="Business Process" /></td>
<td></td>
</tr>
<tr>
<td><strong>Business Function</strong></td>
<td>A <em>business function</em> represents a collection of business behavior based on a chosen set of criteria (typically required business resources and/or competencies), closely aligned to an organization, but not necessarily explicitly governed by the organization.</td>
</tr>
<tr>
<td><img src="image" alt="Business Function" /></td>
<td></td>
</tr>
<tr>
<td><strong>Business Interaction</strong></td>
<td>A <em>business interaction</em> represents a unit of collective business behavior performed by (a collaboration of) two or more business actors, business roles, or business collaborations.</td>
</tr>
<tr>
<td><img src="image" alt="Business Interaction" /></td>
<td></td>
</tr>
<tr>
<td><strong>Business Event</strong></td>
<td>A <em>business event</em> represents an organizational state change.</td>
</tr>
<tr>
<td><img src="image" alt="Business Event" /></td>
<td></td>
</tr>
</tbody>
</table>
### Business Service

A business service represents explicitly defined behavior that a business role, business actor, or business collaboration exposes to its environment.

### Business Object

A business object represents a concept used within a particular business domain.

### Contract

A contract represents a formal or informal specification of an agreement between a provider and a consumer that specifies the rights and obligations associated with a product and establishes functional and non-functional parameters for interaction.

### Representation

A representation represents a perceptible form of the information carried by a business object.

### Product

A product represents a coherent collection of services and/or passive structure elements, accompanied by a contract/set of agreements, which is offered as a whole to (internal or external) customers.
Business Layer Example Diagrams

Business Objects, Representations and Contract

This diagram is concerned with the Business Passive Structure elements and shows how an insurance Claim can be Realized by a number of different representations. An Insurance Policy which is modeled using the a Contract is Realized by a Policy Summary which is a Representation. All of the representations include this element indicated by the use of the Aggregation relationship.

Figure: Example diagram from the ArchiMate Specification showing Business Objects, Representations and Contracts
## Application Layer Elements

### Table of Application Layer Elements

<table>
<thead>
<tr>
<th>Element Type</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Component</td>
<td>An <em>application component</em> represents an encapsulation of application functionality aligned to implementation structure, which is modular and replaceable.</td>
</tr>
<tr>
<td>Application Collaboration</td>
<td>An <em>application collaboration</em> represents an aggregate of two or more application internal active structure elements that work together to perform collective application behavior.</td>
</tr>
<tr>
<td>Application Interface</td>
<td>An <em>application interface</em> represents a point of access where application services are made available to a user, another application component, or a node.</td>
</tr>
<tr>
<td>Application Function</td>
<td>An <em>application function</em> represents automated behavior that can be performed by an application component.</td>
</tr>
<tr>
<td>Application Interaction</td>
<td>An <em>application interaction</em> represents a unit of collective application behavior performed by (a collaboration of) two or more application components.</td>
</tr>
<tr>
<td>Application Process</td>
<td>An <em>application process</em> represents a sequence of application behaviors that achieves a specific result.</td>
</tr>
<tr>
<td>Application Event</td>
<td>An <em>application event</em> represents an application state change.</td>
</tr>
<tr>
<td>Application Service</td>
<td>An <em>application service</em> represents an explicitly defined exposed application behavior.</td>
</tr>
<tr>
<td>Data Object</td>
<td>A data object represents data structured for automated processing.</td>
</tr>
</tbody>
</table>
Application Layer Example Diagrams

Application Components, Interfaces and Collaborations

This diagram demonstrates the use of Application Components, Application Interfaces and Application Collaborations. The Online Travel Insurance Sales Collaboration is aggregated of a Quotation and Purchase system modeled as Application Components. The Collaboration is composed of a Web Service Interface which in turn Serves the Travel Website.

Figure: Example diagram from the ArchiMate Specification showing Application Components, Interfaces and Collaborations
# Technology Layer Elements

## Example 31: Technology Behavior Elements

### Technology Elements

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Node</strong></td>
<td>A <em>node</em> represents a computational or physical resource that hosts, manipulates, or interacts with other computational or physical resources.</td>
</tr>
<tr>
<td><img src="image" alt="Node A" /></td>
<td></td>
</tr>
<tr>
<td><strong>Device</strong></td>
<td>A <em>device</em> represents a physical IT resource upon which system software and artifacts can be stored or deployed for execution.</td>
</tr>
<tr>
<td><img src="image" alt="Device A" /></td>
<td></td>
</tr>
<tr>
<td><strong>System Software</strong></td>
<td><em>System software</em> represents software that provides or contributes to an environment for storing, executing, and using software or data deployed within it.</td>
</tr>
<tr>
<td><img src="image" alt="System Software A" /></td>
<td></td>
</tr>
<tr>
<td><strong>Technology Collaboration</strong></td>
<td>A <em>technology collaboration</em> represents an aggregate of two or more technology internal active structure elements that work together to perform collective technology behavior.</td>
</tr>
<tr>
<td><img src="image" alt="Technology Collaboration A" /></td>
<td></td>
</tr>
<tr>
<td><strong>Technology Interface</strong></td>
<td>A <em>technology interface</em> represents a point of access where technology services offered by a node can be accessed.</td>
</tr>
<tr>
<td><img src="image" alt="Technology Interface A" /></td>
<td></td>
</tr>
<tr>
<td><strong>Path</strong></td>
<td>A <em>path</em> represents a link between two or more nodes, through which these nodes can exchange data, energy, or material.</td>
</tr>
<tr>
<td><img src="image" alt="Path A" /></td>
<td></td>
</tr>
<tr>
<td><strong>Communication Network</strong></td>
<td><em>A communication network</em> represents a set of structures that connects nodes for</td>
</tr>
<tr>
<td><img src="image" alt="Communication Network" /></td>
<td></td>
</tr>
<tr>
<td>Concept</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Communication Network A</td>
<td>Transmission, routing, and reception of data.</td>
</tr>
<tr>
<td>Technology Function</td>
<td>A technology function represents a collection of technology behavior that can be performed by a node.</td>
</tr>
<tr>
<td>Technology Process</td>
<td>A technology process represents a sequence of technology behaviors that achieves a specific result.</td>
</tr>
<tr>
<td>Technology Interaction</td>
<td>A technology interaction represents a unit of collective technology behavior performed by (a collaboration of) two or more nodes.</td>
</tr>
<tr>
<td>Technology Event</td>
<td>A technology event represents a technology state change.</td>
</tr>
<tr>
<td>Technology Service</td>
<td>A technology service represents an explicitly defined exposed technology behavior.</td>
</tr>
<tr>
<td>Artifact</td>
<td>An artifact represents a piece of data that is used or produced in a software development process, or by deployment and operation of an IT system.</td>
</tr>
</tbody>
</table>
Technology Examples

Technology Functions, Interfaces, Processes and Events

This diagram concerns Technology Functions, Interfaces, Processes and Events. The Technology Architect has modeled the Replicate Remote Data Process and the Technology Services and Functions that are required to support the Service. The Database Replication Function is composed of four other Functions that are represented using the 'nested' visualization option. The Composition relationships still exist between each of the four Technology Functions and the Database Replication function they are just suppressed in the diagram.

Figure: Example diagram from the ArchiMate Specification showing Technology Functions, Interfaces, Processes and Events.
## Physical Layer Elements

### Table of Physical Elements

<table>
<thead>
<tr>
<th>Equipment</th>
<th>Equipment represents one or more physical machines, tools, or instruments that can create, use, store, move, or transform materials.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Facility</td>
<td>A Facility represents a physical structure or environment.</td>
</tr>
<tr>
<td>Distribution Network</td>
<td>A Distribution Network represents a physical network used to transport materials or energy.</td>
</tr>
<tr>
<td>Material</td>
<td>Material represents tangible physical matter or energy.</td>
</tr>
</tbody>
</table>
Physical Examples

This diagram is concerned with Physical things that are important to the Architecture, including Facilities, Equipment, Distribution Networks and Material. We see two other elements that the modeler has used; one is Path from the Technology Layer and the other is Location, which is a Composite element. The diagram describes what is being produced at a Manufacturing Plant modeled as a Facility, and the logistics in the form of Shipping of overseas items and trucking for domestic items to move the assembled items to local and national Distribution Centers.

Figure: Example diagram from the ArchiMate Specification showing Facilities, Equipment, Material and Distribution Networks.
# Implementation and Migration Layer Elements

## Table of Implementation and Migration Elements

<table>
<thead>
<tr>
<th>Work Package</th>
<th>A <em>Work Package</em> represents a series of actions identified and designed to achieve specific results within specified time and resource constraints.</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Work Package A" /></td>
<td></td>
</tr>
<tr>
<td>Deliverable</td>
<td>A <em>Deliverable</em> represents a precisely-defined result of a <em>Work Package</em>.</td>
</tr>
<tr>
<td><img src="image" alt="Deliverable A" /></td>
<td></td>
</tr>
<tr>
<td>Implementation Event</td>
<td>An <em>Implementation Event</em> represents a state change related to implementation or migration.</td>
</tr>
<tr>
<td><img src="image" alt="Implementation Event A" /></td>
<td></td>
</tr>
<tr>
<td>Plateau</td>
<td>A <em>Plateau</em> represents a relatively stable state of the architecture that exists during a limited period of time.</td>
</tr>
<tr>
<td><img src="image" alt="Plateau A" /></td>
<td></td>
</tr>
<tr>
<td>Gap</td>
<td>A <em>Gap</em> represents a statement of difference between two plateaus.</td>
</tr>
<tr>
<td><img src="image" alt="Gap A" /></td>
<td></td>
</tr>
</tbody>
</table>
Implementation and Migration Example

Plateaus, Gaps, Deliverables, Work Packages and Events

This diagram shows a number of Plateaus and the Gaps that are Realized by a number of Deliverables, which in turn are realized by a number of Work Packages. In this example the Work Package 'Next Generation Services Program' is composed of three subordinate Work Packages. The first of these - 'Architecture and Planning' - is triggered when the Program Approved Event is fired. This Work Package realizes a series of Deliverables, which in turn realize the Plateau. Gaps are used to record the result of Gap Analysis workshops and are important inputs for migration planning.

Figure: Example diagram from the ArchiMate Specification, showing Plateaus, Gaps, Deliverables, Work Packages and Events.
Composite Elements

Composite elements are special elements that consist of other concepts, from one or - more typically - multiple aspects or layers of the ArchiMate language. There are two composite elements, as described in this table. Composite elements can themselves aggregate or compose other composite elements, leading to a hierarchy of these elements.

Table of Composite Elements

<table>
<thead>
<tr>
<th>Grouping</th>
<th>The <em>Grouping</em> element aggregates or composes concepts that belong together based on some common characteristic.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location</td>
<td><em>Location</em> represents a conceptual or physical place or position where concepts are located (e.g. structure elements) or performed (e.g. behavior elements).</td>
</tr>
</tbody>
</table>
Relationships

Relationships are an important part of the ArchiMate language; they bind elements to other elements and - in some cases - other relationships. Enterprise Architects allows relationships to be created in a number of ways; in this section we will discuss three of the main ways of creating relationships.

Creating Relationships - Toolbox

Relationships can be selected from the diagrams toolbox pages which are specific to the diagram or view that is being constructed. The relationship appear in a toolbox page beneath the elements and can be dragged between any two elements or in some cases between an element and a relationship. The relationships can be used with any pair of elements in the repository.

Creating Relationships - Quick Linker

Relationships can be drawn between two elements or an element and another relationship directly in a diagram, using the element or relationship Quick Linker. The drawing of a relationship is initiated by dragging the small vertical arrow that is positioned at the top right of an element or in the center of a relationship. The allowable elements will be displayed when the linker is drawn to target a diagram object.
Creating Relationships - Matrix

Relationships can be created using the matrix which is a powerful tool for visualizing the relationship between two sets of elements in a spreadsheet like appearance. The matrix displays the two sets of element based on the package location and their type. The type of relationship can be specified which then displays an arrow which indicates pairs of elements one from x-axis and one from the y-axis are related by the specified relationship. A new relationship can be added by right-clicking in an empty cell and selecting 'Create new relationship'.
Relationships Overview

Enterprise Architect is a powerful language-compliant platform that makes it easy to create industry standard Enterprise Architecture diagrams using ArchiMate. The tool allows the modeler to work with relationships flexibly, including being able to change the styles of the connectors while at the same time maintaining the rules and constraints of the ArchiMate language, including restricting what elements can be connected together.

Allowable Element Relationships

The compliance has been achieved by building the ArchiMate facility on top of a flexible and compliant meta-model that implements the relationship table specified in Appendix B of the ArchiMate Specification. This ensures that if the specification changes it is simply a matter of updating the metamodel and all aspects of the tool will reflect the change. This makes it easy for the user to create first-class and compliant models as the interface restricts the available relationships when attempting to draw a connection between two elements in a diagram canvas.

In this example we see a modeler using the Quick Linker (a fast way of connecting elements) to create a relationship from a Business Actor to a Business Function. Once the relationship is dropped onto the target a menu is opened that displays the allowed relationships between the two element types based on the direction of the connection.

In this diagram we see that the user has selected the assignment connector, as this is an allowed relationship from a Business Actor to a Business Function.
Changing Size Colour and Style

Any of the standard ArchiMate relationships can be changed to different sizes and colors without affecting the meaning or syntax of the diagrams. Enterprise Architect provides great flexibility for changing the relationships and elements, and the diagram styles and themes. This illustration shows two different diagrams where the same relationship has been rendered in a different line width and color.

The options to change the style of a relationship are available from a number of locations, including the ribbons and the element Quick Style icon that is displayed when the relationship's context menu is selected in the diagram. This illustration shows the options available from the diagram.

Nesting

Enterprise Architect allows elements to be nested as an alternative way of representing relationships. This is a powerful and compact notation for some diagrams, alleviating the issue of some stakeholders finding the ArchiMate visual syntax difficult to interpret. It does, however, introduce ambiguities when the diagrams are viewed outside the tool. This is an issue with the ArchiMate language itself rather than being any limitation of the tool, and while the diagrams are being viewed inside Enterprise Architect the ambiguities can always be resolved. Enterprise Architect allows nested views to be created and visualized, providing a number of ways to remove the ambiguities by use of other windows that can be opened in the workspace; these clearly indicate the type of relationship implied by the element nesting.

To create nested elements, the relationships between elements must already exist or first be created. A modeler can then use drag-and-drop to nest one or more elements inside another element. Enterprise Architect hides the relationships in...
the diagram but they are visible through a number of other workspace windows, including the Traceability window, the Relationships window and the 'Details' tab of the Inspector window. This illustration shows the Traceability window when the containing element is selected, clearly showing the Composition relationships that exist in the model; these are not displayed as direct relationships in the diagram but are indicated by the nesting.

**Connecting to Other Relationships**

Enterprise Architect, in compliance with the ArchiMate specification, allows you to connect one end of certain relationships to other relationships. This is powerful device to express particular modeling concepts; for example, the data that is transmitted between two elements such as two Application Components or Business Functions. This diagram shows two examples where this feature has been used.
Structural Relationships

Structural relationships depict the “static” coherence within an architecture. There is always an element at the uniting end (from end) of the relationship, for the assignment and realization relationships it can be an element or a relationships connector. The uniting end (to end) of the relationship typically connects to an element but in some circumstances it can be another relationship or relationship connector. These relationships are not only about how things are structured but also about responsibility for the performance of behavior, storage, or execution or the representation of how a more concrete element plays a critical role in the creation, achievement, sustenance, or operation. This table describes the structural relationships.

<table>
<thead>
<tr>
<th>Structural Relationships</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Composition</strong></td>
</tr>
<tr>
<td><img src="Diagram1" alt="Diagram" /></td>
</tr>
<tr>
<td>The <em>composition</em> relation represent that an element consists of one or more other concepts.</td>
</tr>
<tr>
<td><strong>Aggregation</strong></td>
</tr>
<tr>
<td><img src="Diagram2" alt="Diagram" /></td>
</tr>
<tr>
<td>The <em>aggregation</em> relation represent that an element combines one or more other concepts.</td>
</tr>
<tr>
<td><strong>Assignment</strong></td>
</tr>
<tr>
<td><img src="Diagram3" alt="Diagram" /></td>
</tr>
<tr>
<td>The <em>assignment</em> relation represent the allocation of responsibility, performance of behavior, storage, or execution.</td>
</tr>
<tr>
<td><strong>Realization</strong></td>
</tr>
<tr>
<td><img src="Diagram4" alt="Diagram" /></td>
</tr>
<tr>
<td>The <em>realization</em> relation represent that an entity plays a critical role in the creation, achievement, sustenance, or operation of a more abstract entity.</td>
</tr>
</tbody>
</table>
Dependency Relationships

Dependency relationships describe how elements support or are used by other elements. They can indicated direction in the form of an arrowhead at one end of the line - the direction is optional for some relationships () and mandatory for others (Influence). Four types of dependency relationship are distinguished as indicated in this table.

**Dependency Relationships**

<table>
<thead>
<tr>
<th>Serving</th>
<th>The <em>serving</em> relationship represents that an element provides its functionality to another element.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>The <em>access</em> relationship represents the ability of behavior and active structure elements to observe or act upon passive structure elements.</td>
</tr>
<tr>
<td>Influence</td>
<td>The <em>influence</em> relationship represents that an element affects the implementation or achievement of some motivation element.</td>
</tr>
<tr>
<td>Association</td>
<td>An <em>association</em> relationship represents an unspecified relationship, or one that is not represented by another ArchiMate relationship.</td>
</tr>
</tbody>
</table>
Dynamic Relationships

The dynamic relationships describe temporal dependencies between elements within the architecture. They are often mistakenly interpreted as causal relationships implying that the element at the tail end of the relationship somehow causes the element at the arrow end to 'fire' - this is not correct. In the case of the Flow relationship an item can be transmitted between the two connected elements. Two types of dynamic relationship are distinguished as indicated in this table.

## Dynamic Relationships

<table>
<thead>
<tr>
<th>Triggering</th>
<th>The <em>triggering</em> relationship represents a temporal or causal relationship between elements.</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Diagram" /></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Flow</th>
<th>The <em>flow</em> relationship represents transfer from one element to another.</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Diagram" /></td>
<td></td>
</tr>
</tbody>
</table>

(c) Sparx Systems 2020
Other Relationships

The other relationships category is a catch all label for relationships that don't fit into the other categories. There is only one relationship in this category entitled Specialization which is a directed relationship of family meaning that the element at the tail end is a more specific than the element at the arrow-end of the relationship.

**Other Relationships**

<table>
<thead>
<tr>
<th>Specialization</th>
<th>The <em>specialization</em> relationship represents that an element is a particular kind of another element.</th>
</tr>
</thead>
</table>
Relationship Connectors

A junction is used in a number of situations to connect relationships of the same type. A path with junctions that connect relationships of this type is only allowed between two concepts, if a direct relationship of that type between these concepts is also permitted. Simply put, you cannot use junctions to create relationships between concepts that would otherwise not be allowed.

Connectors

<table>
<thead>
<tr>
<th>Connector</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Junction (AND)</td>
<td>A junction is used to connect relationships of the same type to indicate that all connectors on the multiple side(s) of the junction will be invoked.</td>
</tr>
<tr>
<td>Junction (OR)</td>
<td>A junction is used to connect relationships of the same type to indicate that one (and only one) connector on the multiple side(s) of the junction will be invoked.</td>
</tr>
</tbody>
</table>
Alignment between Layers

The ArchiMate language is divided into a number of layers which is a useful mechanism to separate the concerns of various disciplines (e.g. Business from Application layers). This creates a division in the models that are created and so we need a mechanism to bridge this separation so that we can ensure that the elements and relationships created in one layer are able to be visualized in relation to the elements in the layer above or below. (e.g. elements in the business layer should be able to connect to elements in the strategic layer above and the application layer below.)

ArchiMate has two important relationships that are useful for this purpose.

Serving Relationship

Serving relationships can be used to connect an application layer element to a business layer element for example between an application service and the different types of business behavior elements such as business processes and functions, and between application interfaces and business roles. The relationships can also be drawn in the opposite direction between business services and application behavior elements, and between business interface and application component. These relationships represent the behavioral and structural aspects of the support of the business by applications.

Realization Relationship

Realization relationships can be used to connect an application layer element to a business layer element for example an application process or function can be said to realize a business process or function, or a data object or a technology object can be said to realize a business object, so as to indicate that the data object is a digital representation of the corresponding business object, or the technology object is a physical representation of the business object. The relationship is always drawn in the direction from the lower layer with the arrow pointing to the higher layer.
Customizing the ArchiMate Language

Enterprise Architect provides a powerful capability to customize language using its profile system which is part of a sophisticated feature to create Model Driven Generation extensions to the tool. These extensions can be designed and built within the tool and the resulting technology can be imported into the tool making customizations available to Enterprise Architects and other repository users.

A typical usage scenario would be that a team might want to categorize their Application Portfolio based on a range of criteria and be able to analyze architectures based on the applications to answer questions like:

- What are the set of technologies underpinning our mission critical systems
- Which applications run on a cloud platform
- Which Business Capabilities would be compromised if Vendor A became insolvent

This would require the provision and maintenance of properties for each application. This can be done using the profile system where any number of properties can be added to an ArchiMate element or connector and the customized element can be added to custom toolboxes. The elements and connectors are then available to the architect for modeling and once the properties are populated with organization specific data reports and visualizations can be created that use the properties.

In this diagram we see the customization of an ArchiMate Application Component as part of the creation of a profile. We see that the user defined Application Component is a specialization of the standard ArchiMate element (indicated by the triangular arrow-headed relationship). We also see that the specialized element has a number of added properties such as Business Owner, Technical Owner and Deployment Type. These are user defined properties and provide the mechanism for adding organization specific information that will help in an architectures analysis to answer questions like those presented earlier.

Changing Element Default Appearance

Enterprise Architect allows the appearance of elements in diagrams to be altered to suit the intended audience. When
creating a profile and a customized element it is also possible to set its default appearance so when it is added to any diagram it will take on that appearance. For example you might want to use a different set of colors that has meaning in your architecture practice. This illustration shows the mechanism for setting the default color, font and other presentation options.

Changing an Element's Shape

While the standard ArchiMate element shapes are important for communicating to audiences, there are situations where you might want to change the shape of an element that you have customized as part of a profile. For example, an organization is a vendor of 'Smart Phones' and wants to represent a product with an icon resembling the phone. Alternatively, an architect wants to represent a network using a Cloud as an alternative to a standard Technology element icon.

Shape Script Example

You can create a wide range of shapes, effects and text statements using Shape Scripts, to enhance the appearance and information value of the elements and connectors you create. An example of a script is provided in this table.

<table>
<thead>
<tr>
<th>Shape</th>
<th>Example Script</th>
</tr>
</thead>
</table>
| ![Cloud Path Example Shape](image) | // CLOUD PATH EXAMPLE SHAPE  
shape main  
{  
    StartCloudPath();  
    Rectangle(0, 0, 100, 100);  
    EndPath();  
    FillAndStrokePath();  
} |

Changing Connector Default Appearance
Enterprise Architect allows the appearance of connectors in diagrams to be altered to suit the intended audience. When creating a profile and a customized connector it is also possible to set its default appearance so when it is added to any diagram it will take on that appearance. For example you might want to use a different set of colors that has meaning in your architecture practice. This illustration shows the mechanism for setting the default color, font and other presentation options.

Creating Proprietary Elements and Relationships

In addition to customizing existing ArchiMate elements as described earlier, it is possible to add completely new and proprietary elements and connectors, as long as they don't have any conflicting or obstructing meaning with respect to the corpus of ArchiMate elements and customizations. An example of this is in the Security Architecture Domain, where an architect might want to have an element that represents a Security Policy.
Example Profile

Enterprise Architect is a highly compliant tool for modeling enterprise architectures using the ArchiMate language and in compliance with the specification it allows users, teams, organizations or industries to customize the language. This is always intended as a specialization of the existing language constructs and is never intended to compromise or compete with the linguistic structure of the language. This is a very useful feature and provides a mechanism for the addition of user-defined properties to elements, relationships and relationship connectors. For example engineering organizations may wish to add user defined properties associated with safety or performance whereas a financial organization may wish to add properties concerning money or credit.

The following example shows the specialization of an ArchiMate business actor into two elements and an ArchiMate Flow relationship into a Money Flow. The Organizational Unit for example has two user defined properties added namely:

- Corporate Level
- Location

```
threat agent
+ _metatype = Threat Agent
+ Level
+ Type

organizational unit
+ _metatype = Organizational Unit
+ Corporate Level
+ Location

money flow
+ _metatype = Money Flow
+ Amount
+ Currency = NZD
```

Figure: Showing the specialization of an ArchiMate business actor and a flow relationship
Exchanging ArchiMate Models

Enterprise Architect supports model exchange in accordance with the specification and allows architects and others to export and import models that comply with the exchange file format. Enterprise Architect is a powerful tool and its price point, flexible licensing model and browser-based access (to the Pro Cloud Server) allows the tool to be made available to a large number of users from interdisciplinary teams.

The tool provides these two exchange options:

- Generate Model Exchange File - allowing models to be exported from Enterprise Architect
- Import Model Exchange File - allowing models to be imported into Enterprise Architect
Generate Model Exchange File

After you have created an ArchiMate® 2 or ArchiMate® 3.1 model using the built-in MDG Technology for ArchiMate 2 or ArchiMate 3.1 respectively, you can generate a Model Exchange File from the model Package.

Access

<table>
<thead>
<tr>
<th>Ribbon</th>
<th>Specialize &gt; Technologies &gt; ArchiMate &gt; Generate Model Exchange File</th>
</tr>
</thead>
<tbody>
<tr>
<td>Context Menu</td>
<td>Right-click on Package</td>
</tr>
</tbody>
</table>

Generate Model Exchange File

<table>
<thead>
<tr>
<th>Option</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Package</td>
<td>Displays the name of the currently-selected Package. If this is not the required Package, click on the button and select the correct Package.</td>
</tr>
<tr>
<td>Filename</td>
<td>Type in the file path and name of the file to be generated, or click on the button and browse for the location.</td>
</tr>
<tr>
<td>Version</td>
<td>Click on the drop-down arrow and select the version of the Model Exchange File.</td>
</tr>
<tr>
<td>Language</td>
<td>(Optional) Select the language identifier for the content of the elements in the generated file.</td>
</tr>
</tbody>
</table>
| Include | Select the appropriate checkboxes:  
- 'Element Relationships' - to include connectors between the elements being generated  
- 'Tagged Values' - to include the Tagged Values for each element and relationship being generated  
- 'Element Organization' - to include the structural organization of the elements inside the selected Package  
- 'Diagrams' - include the ArchiMate diagrams inside the selected Package |
| Generate | Click on this button to generate the Model Exchange File.  
The progress of the file generation is reported in the 'ArchiMate' tab of the System Output window. A message box also displays to indicate when the generation is complete; click on the OK button to clear the message. |
| View File | Click on this button to display the contents of the generated file. |
| Close | Click on this button to close this 'Generate Model Exchange File Format' dialog. |
Notes

- This facility is available in the Professional, Corporate, Unified and Ultimate Editions of Enterprise Architect
- The 'Language' field displays the 2-letter codes from the Language Subtag Registry: http://www.iana.org/assignments/language-subtag-registry/language-subtag-registry
- The Model Exchange File generated by Enterprise Architect adheres to The Open Group Specified Schema
- Generated elements and diagrams will be displayed in the System Output window – double-click on an item in the System Output window to highlight it in the Browser window
- Elements, connectors and diagrams that are not from the built-in MDG Technology for ArchiMate 2 or ArchiMate 3.1 will be ignored during generation
- The Network and Communication Path connectors from the 'ArchiMate 2::Technology' toolbox in Enterprise Architect will not currently be generated as they do not have any mapping in The Open Group Specified Schema
- The Technology <object> elements, Path connector, Communication Network connector and Distribution Network connector from the 'ArchiMate 3.1 Technology' toolbox in Enterprise Architect will not currently be generated as they do not have any mapping in The Open Group Specified Schema
Import ArchiMate Model Exchange File

If you have a valid ArchiMate® 2 or ArchiMate® 3.1 Model Exchange File, you can import it into your Enterprise Architect project as a UML model.

Access

<table>
<thead>
<tr>
<th>Ribbon</th>
<th>Specialize &gt; Technologies &gt; ArchiMate &gt; Import Model Exchange File</th>
</tr>
</thead>
<tbody>
<tr>
<td>Context Menu</td>
<td>Right-click on Package</td>
</tr>
</tbody>
</table>

Import Model Exchange File

<table>
<thead>
<tr>
<th>Option</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Package</td>
<td>Displays the name of the currently-selected Package.</td>
</tr>
<tr>
<td></td>
<td>If this is not the required Package, click on the button to select the Package Browser and select the correct Package.</td>
</tr>
<tr>
<td>Filename</td>
<td>Type in the file path of the file to be imported, or click on the button and browse for the location.</td>
</tr>
<tr>
<td>Connector Style</td>
<td>Select one of these styles to set on the imported connectors:</td>
</tr>
<tr>
<td></td>
<td>• Orthogonal – Rounded (rounded corners)</td>
</tr>
<tr>
<td></td>
<td>• Orthogonal – Square (square corners)</td>
</tr>
<tr>
<td></td>
<td>• Custom</td>
</tr>
<tr>
<td>View File</td>
<td>Click on this button to open and view the file to be imported.</td>
</tr>
<tr>
<td>Import</td>
<td>Click on this button to import the selected Model Exchange file into the specified Package.</td>
</tr>
<tr>
<td></td>
<td>The progress of the file import is reported in the 'ArchiMate' tab of the System Output window. A message box also displays to indicate when the import is complete; click on the OK button to clear the message.</td>
</tr>
<tr>
<td>Close</td>
<td>Click on this button to close the dialog.</td>
</tr>
<tr>
<td>Help</td>
<td>Click on this button to display this Help topic.</td>
</tr>
</tbody>
</table>

Notes

- This facility is available in the Professional, Corporate, Unified and Ultimate Editions of Enterprise Architect
• The Model Exchange File should adhere to The Open Group Specified Schema
• Elements, connectors and diagrams that are not specified in *The Open Group Schema* will be ignored during import
• Imported elements and diagrams will be displayed in the System Output window – double-click on an item in this window to highlight it in the Browser window
Migrate to the Latest ArchiMate Version

If you have created a model under one version of ArchiMate, you can migrate all or part of it to a later version; for example, migrate the model from ArchiMate 2 to ArchiMate 3.1. You perform the migration using the Automation Project Interface function Migrate(), which updates the Tagged Values and, if required, stereotypes to the later version of ArchiMate for all elements, attributes, connectors and diagrams under the selected Package or element.

When you migrate your model, you must have both releases of the Technology enabled; when the migration is complete, disable the older release of the Technology.

Migrate ArchiMate 2 model to ArchiMate 3.1

The MDG Technology for ArchiMate 3.1 has a migration script built into it, to upgrade a model built for ArchiMate 2 to ArchiMate 3.1. To execute this script:

1. Select the 'Specialize > Tools > Scripting' ribbon option.
2. Expand the ArchiMate® 3.1 folder.
3. Right-click on the 'Migrate ArchiMate 2 to ArchiMate 3' script.
4. Click on the 'Run Script' option from the context menu.
5. Monitor the execution of the script in the System Output window.

When the script has finished executing, disable the ArchiMate 2 technology.

Notes (Migrate ArchiMate 2 to ArchiMate 3.1)

- If you want to migrate from an ArchiMate model to an ArchiMate 3.1 model, you must do so in two stages: from ArchiMate to ArchiMate 2, and from ArchiMate 2 to ArchiMate 3.1

Migrate ArchiMate model to ArchiMate 2

You must create a script to call the Migrate() function to migrate a Package or element to ArchiMate 2: this is a VB script example of such a script:

```vba
Sub MigrateElement (sGUID, lngPackageID)
    Dim proj as EA.Project
    set proj = Repository.GetProjectInterface
    proj.Migrate sGUID, "ArchiMate", "ArchiMate 2"
    'refresh the model
    If lngPackageID<>0 Then
        Repository.RefreshModelView (lngPackageID)
    End If
End Sub
Sub MigrateSelectedItem
    Dim selType
    Dim selElement as EA.Element
    Dim selPackage as EA.Package
```
selType = GetTreeSelectedItemType
If selType = 4 Then 'means Element
    set selElement = GetTreeSelectedObject
    MigrateElement selElement.ElementGUID, selElement.PackageID
    MsgBox "Element Migration Completed",0,"ArchiMate 2 Migration"
ElseIf selType = 5 Then 'means Package
    set selPackage = GetTreeSelectedObject
    MigrateElement selPackage.PackageGUID, selPackage.PackageID
    MsgBox "Package Migration Completed",0,"ArchiMate 2 Migration"
Else
    MsgBox "Select a Package or Element in the Browser window to initiate migration",0,"ArchiMate 2 Migration"
End If
End Sub
Sub Main
    MigrateSelectedItem
End Sub
Main

Notes (Migrate ArchiMate to ArchiMate 2)

- All diagrams are converted to Business Layer diagrams
- Collaboration elements are converted to Business Collaboration or Application Collaboration depending on the value of the 'collaborationtype' Tagged Value
- Object elements are converted to Business Object, Contract or Data Object depending on the value of the 'objecttype' Tagged Value
- Interface elements are converted to Business Interface, Application Interface or Infrastructure Interface depending on the value of the 'interfacetype' Tagged Value
- Function elements are converted to Business Function or Application Function depending on the value of the 'functiontype' Tagged Value
- Interaction elements are converted to Business Interaction or Application Interaction depending on the value of the 'interactiontype' Tagged Value
- Service elements are converted to Business Service, Application Service or Infrastructure Service depending on the value of the 'servicetype' Tagged Value
- The 'iconstyle' Tagged Value is removed and the 'Use Rectangle Notation' menu option is set on or off as appropriate
- Process elements become Business Process elements
- Event elements become Business Event elements
- Actor elements become Business Actor elements
- Role elements become Business Role elements
- Component elements become Application Component elements
- Software elements become System Software elements
- Specialisation connectors become Specialization connectors
- Realisation connectors become Realization connectors
- Network Connector connectors become Network
- Node elements now extend UML Class elements
- Software elements now extend UML Class elements
- Junction elements now extend UML Decision elements