THE OPEN GROUP SERIES

ArchiMate® 2.1 Specification

Van Haren PUBLISHING

THE Open GROUP

ArchiMate® 2.1 Specification

The Open Group Publications available from Van Haren Publishing

The TOGAF Series:
TOGAF® Version 9.1
TOGAF® Version 9.1 – A Pocket Guide
TOGAF® 9 Foundation Study Guide, 3rd Edition
TOGAF® 9 Certified Study Guide, 3rd Edition

The Open Group Series:
Cloud Computing for Business – The Open Group Guide
ArchiMate® 2.1 – A Pocket Guide
ArchiMate® 2.1 Specification
ArchiMate® 2 Certification – Study Guide

The Open Group Security Series:
Open Information Security Management Maturity Model (O-ISM3)
Open Enterprise Security Architecture (O-ESA)
Risk Management – The Open Group Guide
The Open FAIR™ Body of Knowledge – A Pocket Guide

All titles are available to purchase from:
www.opengroup.org
www.vanharen.net
and also many international and online distributors.

Open Group Standard

ArchiMate® 2.1 Specification

Title: **ArchiMate® 2.1 Specification**

A Publication of: The Open Group

Publisher: Van Haren Publishing, Zaltbommel, www.vanharen.net

ISBN Hard copy: 978 94 018 0003 7
ISBN eBook: 978 94 018 0508 7
ISBN ePub: 978 94 018 0509 4

Edition: First edition, second impression, June 2012
First edition, second impression, March 2013
Second edition, first impression, December 2013
Second edition, second impression, May 2015

Design and Layout: CO2 Premedia, Amersfoort-NL

Document Number: C13L

Comments relating to the material contained in this document may be submitted to:
The Open Group
Apex Plaza
Forbury Road
Reading
Berkshire, RG1 1AX
United Kingdom
or by electronic mail to: ogspecs@opengroup.org

Contents

Table of Figures X
Preface XIII
Trademarks XVI
Acknowledgements XVII
Referenced Documents XIX

1 Introduction 1

2 Language Structure 3
2.1 Design Approach 3
2.2 Core Concepts 4
2.3 Collaboration and Interaction 7
2.4 Relationships 7
2.5 Layering 8
2.6 The ArchiMate Framework 8
2.7 Motivation Extension 11
2.8 Implementation and Migration Extension 13
2.9 ArchiMate and TOGAF 15
2.10 Use of Colors 16

3 Business Layer 19
3.1 Business Layer Metamodel 19
3.2 Active Structure Concepts 20
3.2.1 Business Actor 20
3.2.2 Business Role 22
3.2.3 Business Collaboration 23
3.2.4 Business Interface 24
3.2.5 Location 26
3.3 Behavioral Concepts 27
3.3.1 Business Process 28
3.3.2 Business Function 30
3.3.3 Business Interaction 32
3.3.4 Business Event 33
3.3.5 Business Service 35
3.4 Passive Structure Concepts 36

3.4.1 Business Object 38
3.4.2 Representation 39
3.4.3 Meaning. 41
3.4.4 Value 42
3.4.5 Product 43
3.4.6 Contract. 45
3.5 Summary of Business Layer Concepts. 47

4 Application Layer 49
4.1 Application Layer Metamodel. 49
4.2 Active Structure Concepts. 50
4.2.1 Application Component 50
4.2.2 Application Collaboration 52
4.2.3 Application Interface 53
4.3 Behavioral Concepts. 54
4.3.1 Application Function. 55
4.3.2 Application Interaction 56
4.3.3 Application Service 57
4.4 Passive Structure Concepts 59
4.4.1 Data Object. 59
4.5 Summary of Application Layer Components 61

5 Technology Layer 63
5.1 Technology Layer Metamodel. 63
5.2 Active Structure Concepts. 63
5.2.1 Node. 64
5.2.2 Device 65
5.2.3 System Software 67
5.2.4 Infrastructure Interface. 68
5.2.5 Network. 69
5.2.6 Communication Path 70
5.3 Behavioral Concepts. 71
5.3.1 Infrastructure Function. 71
5.3.2 Infrastructure Service 72
5.4 Passive Structure Concepts 73
5.4.1 Artifact 73
5.5 Summary of Technology Layer Concepts 75

6 Cross-Layer Dependencies **77**
6.1 Business Layer and Lower Layers Alignment 77
6.2 Application-Technology Alignment 78

7 Relationships **81**
7.1 Structural Relationships. 81
7.1.1 Composition Relationship 81
7.1.2 Aggregation Relationship 82
7.1.3 Assignment Relationship 83
7.1.4 Realization Relationship 84
7.1.5 Used By Relationship. 85
7.1.6 Access Relationship 86
7.1.7 Association Relationship. 87
7.2 Dynamic Relationships 88
7.2.1 Triggering Relationship. 88
7.2.2 Flow Relationship. 89
7.3 Other Relationships 90
7.3.1 Grouping. 90
7.3.2 Junction. 91
7.3.3 Specialization Relationship. 91
7.4 Summary of Relationships. 92
7.5 Derived Relationships 94

8 Architecture Viewpoints **97**
8.1 Introduction 97
8.2 Views, Viewpoints, and Stakeholders 99
8.3 Viewpoint Classification 100
8.4 Standard Viewpoints in ArchiMate 103
8.4.1 Introductory Viewpoint 104
8.4.2 Organization Viewpoint 106
8.4.3 Actor Co-operation Viewpoint 107
8.4.4 Business Function Viewpoint. 109
8.4.5 Business Process Viewpoint 111
8.4.6 Business Process Co-operation Viewpoint. 112
8.4.7 Product Viewpoint. 114
8.4.8 Application Behavior Viewpoint 116
8.4.9 Application Co-operation Viewpoint 117
8.4.10 Application Structure Viewpoint. 119

8.4.11 Application Usage Viewpoint 121
8.4.12 Infrastructure Viewpoint 123
8.4.13 Infrastructure Usage Viewpoint 124
8.4.14 Implementation and Deployment Viewpoint 126
8.4.15 Information Structure Viewpoint 128
8.4.16 Service Realization Viewpoint 130
8.4.17 Layered Viewpoint 131
8.4.18 Landscape Map Viewpoint 134

9 Language Extension Mechanisms 137
9.1 Adding Attributes to ArchiMate Concepts and Relationships 137
9.2 Specialization of Concepts and Relationships 139

10 Motivation Extension 141
10.1 Motivation Aspect Metamodel 141
10.2 Motivational Concepts 141
10.2.1 Stakeholder 142
10.2.2 Driver 143
10.2.3 Assessment 144
10.2.4 Goal 146
10.2.5 Requirement 147
10.2.6 Constraint 149
10.2.7 Principle 150
10.2.8 Summary of Motivational Concepts 152
10.3 Relationships 153
10.3.1 Association Relationship 153
10.3.2 Aggregation Relationship 154
10.3.3 Realization Relationship 155
10.3.4 Influence Relationship 156
10.3.5 Summary of Relationships 158
10.4 Cross-Aspect Dependencies 159
10.5 Viewpoints 159
10.5.1 Stakeholder Viewpoint 160
10.5.2 Goal Realization Viewpoint 162
10.5.3 Goal Contribution Viewpoint 163
10.5.4 Principles Viewpoint 165
10.5.5 Requirements Realization Viewpoint 166
10.5.6 Motivation Viewpoint 167

11 Implementation and Migration Extension 169
11.1 Implementation and Migration Extension Metamodel 169
11.2 Implementation and Migration Concepts 169
11.2.1 Work Package 169
11.2.2 Deliverable 170
11.2.3 Plateau 171
11.2.4 Gap 172
11.2.5 Summary of Implementation and Migration Concepts 173
11.3 Relationships 174
11.4 Cross-Aspect Dependencies 174
11.5 Viewpoints 176
11.5.1 Project Viewpoint 176
11.5.2 Migration Viewpoint 178
11.5.3 Implementation and Migration Viewpoint 180

12 Future Directions (Informative) 183
12.1 Extending and Refining the Concepts 183
12.1.1 Business Policies and Rules 184
12.1.2 Design Process 184
12.1.3 Other Improvements 184

A Summary of Language Notation (Informative) 185
B Relationship Tables 187

Index 195

Table of Figures

Figure 1: Metamodels at Different Levels of Specificity 4
Figure 2: Generic Metamodel: The Core Concepts of ArchiMate 5
Figure 3: Collaboration and Interaction. 7
Figure 4: Architectural Framework. 9
Figure 5: Relationship between Core and Motivational Elements in ArchiMate 12
Figure 6: Relationships between Motivational, Core, and Implementation and Migration Elements 14
Figure 7: Correspondence between ArchiMate and TOGAF 15
Figure 8: Correspondence between ArchiMate (including extensions) and TOGAF 16
Figure 9: Business Layer Metamodel 19
Figure 10: Business Actor Notation 21
Figure 11: Business Role Notation 22
Figure 12: Business Collaboration Notation 24
Figure 13: Business Interface Notation. 25
Figure 14: Location Notation. 26
Figure 15: Business Process Notation. 29
Figure 16: Business Function Notation 30
Figure 17: Business Interaction Notation. 32
Figure 18: Business Event Notation 34
Figure 19: Business Service Notation 35
Figure 20: Business Object Notation 39
Figure 21: Representation Notation 40
Figure 22: Meaning Notation. 41
Figure 23: Value Notation 43
Figure 24: Product Notation 44
Figure 25: Contract Notation. 45
Figure 26: Application Layer Metamodel. 49
Figure 27: Application Component Notation 51
Figure 28: Application Collaboration Notation 52
Figure 29: Application Interface Notation 54
Figure 30: Application Function Notation. 55
Figure 31: Application Interaction Notation 57
Figure 32: Application Service Notation 58

Figure 33: Data Object Notation 60
Figure 34: Technology Layer Metamodel. 63
Figure 35: Node Notation. 65
Figure 36: Device Notation 66
Figure 37: System Software Notation 67
Figure 38: Infrastructure Interface Notations 68
Figure 39: Network Notation, as Connection and as Box 69
Figure 40: Communication Path Notation, as Connection and as Box. 70
Figure 41: Infrastructure Function Notation. 71
Figure 42: Infrastructure Service Notation 73
Figure 43: Artifact Notation. 74
Figure 44: Relationships between Business Layer and Lower Layer Concepts 78
Figure 45: Relationships between Application Layer and Technology Layer Concepts 79
Figure 46: Composition Notation 81
Figure 47: Aggregation Notation. 82
Figure 48: Assignment Notation 83
Figure 49: Realization Notation. 84
Figure 50: Used By Notation 85
Figure 51: Access Notation. 86
Figure 52: Association Notation 87
Figure 53: Triggering Notation 88
Figure 54: Flow Notation 89
Figure 55: Grouping Notation 90
Figure 56: Junction Notation 91
Figure 57: Specialization Notation 92
Figure 58: Conceptual Model of Architectural Description (from [1]). 99
Figure 59: Classification of Enterprise Architecture Viewpoints 102
Figure 60: Examples of Specialized Concepts and Relationships 139
Figure 61: Motivation Extension Metamodel 141
Figure 62: Stakeholder Notation 143
Figure 63: Driver Notation. 143
Figure 64: Assessment Notation 145
Figure 65: Goal Notation 146
Figure 66: Requirement Notation 148
Figure 67: Constraint Notation 149
Figure 68: Principle Notation 151

Figure 69: Association Notation 153
Figure 70: Aggregation Notation. 154
Figure 71: Realization Notation. 155
Figure 72: Influence Notation 157
Figure 73: Relationships between Motivation Extension and the ArchiMate Core Concepts 159
Figure 74: Implementation and Migration Extension Metamodel. 169
Figure 75: Work Package Notation 170
Figure 76: Deliverable Notation 170
Figure 77: Plateau Notation 172
Figure 78: Gap Notation 173
Figure 79: Relationships between Implementation & Migration Extension and the ArchiMate Core Concepts 174
Figure 80: Relationships between Plateau, Project Result, and Motivation Concepts 176

Preface

The Open Group

The Open Group is a global consortium that enables the achievement of business objectives through IT standards. With more than 400 member organizations, The Open Group has a diverse membership that spans all sectors of the IT community – customers, systems and solutions suppliers, tool vendors, integrators, and consultants, as well as academics and researchers – to:

- Capture, understand, and address current and emerging requirements, and establish policies and share best practices
- Facilitate interoperability, develop consensus, and evolve and integrate specifications and open source technologies
- Offer a comprehensive set of services to enhance the operational efficiency of consortia
- Operate the industry's premier certification service

Further information on The Open Group is available at www.opengroup.org.

The Open Group publishes a wide range of technical documentation, most of which is focused on development of Open Group Standards and Guides, but which also includes white papers, technical studies, certification and testing documentation, and business titles. Full details and a catalog are available at www.opengroup.org/bookstore.

Readers should note that updates – in the form of Corrigenda – may apply to any publication. This information is published at www.opengroup.org/corrigenda.

This Document

This document is the ArchiMate 2.1 Specification, an Open Group Standard.

Issue 2.1 is a maintenance update to ArchiMate 2.0, addressing comments raised since the introduction of ArchiMate 2.0 in 2012. It retains the major features and structure of ArchiMate 2.0, thereby preserving existing investment in ArchiMate 2.0. ArchiMate 2.0 includes a number of corrections, clarifications, and improvements compared to ArchiMate 1.0,

as well as two optional language extensions: the Motivation extension and the Implementation and Migration extension.

Intended Audience

The intended audience of this Technical Standard is threefold:

- Enterprise architecture practitioners, such as architects (application, information, process, infrastructure, products/services, and, obviously, enterprise architects), senior and operational management, project leaders, and anyone committed to work within the reference framework defined by the enterprise architecture. It is assumed that the reader has a certain skill level and is effectively committed to enterprise architecture. Such a person is most likely the architect – that is, someone who has affinity with modeling techniques, knows his way around the organization, and is familiar with information technology.
- Those who intend to implement the ArchiMate language in a software tool. They will find a complete and detailed description of the language in this document.
- The academic community, on which we rely for amending and improving the language based on state-of-the-art research results in the architecture field.

Structure

The structure of this Technical Standard is as follows:

- Chapter 1, Introduction, provides a brief introduction to the purpose of this standard.
- Chapter 2, Language Structure, presents some general ideas, principles, and assumptions underlying the development of the ArchiMate metamodel and introduces the ArchiMate Framework.
- Chapter 3, Business Layer, covers the definition and usage of the business layer concept, together with examples.
- Chapter 4, Application Layer, covers the definition and usage of the application layer concept, together with examples.
- Chapter 5, Technology Layer, covers the definition and usage of the technical infrastructure layer concept, together with examples.
- Chapter 6, Cross-Layer Dependencies, and Chapter 7, Relationships, cover the definition of relationship concepts in a similar way.
- Chapter 8, Architecture Viewpoints, presents and clarifies a set of

architecture viewpoints, developed in ArchiMate based on practical experience. All ArchiMate viewpoints are described in detail. For each viewpoint the comprised concepts and relationships, the guidelines for the viewpoint use, and the goal and target group and of the viewpoint are specified. Furthermore, each viewpoint description contains example models.

- Chapter 9, Language Extension Mechanisms, handles extending and/or specializing the ArchiMate language for specialized or domain-specific purposes.
- Chapter 10, Motivation Extension, describes an optional language extension with concepts, relationships, and viewpoints for expressing the motivation for an architecture (e.g., stakeholders, concerns, goals, principles, and requirements).
- Chapter 11, Implementation and Migration Extension, describes an optional language extension with concepts, relationships, and viewpoints for expressing the implementation and migration aspects of an architecture (e.g., project, programs, plateaus, and gaps).
- Chapter 12, Future Directions, is an informative chapter that identifies extensions and directions for developments in the next versions of the language.
- Appendix A, Summary of Language Notation (Informative), is an informative appendix.
- Appendix B, Relationship Tables, is a normative appendix detailing the required relationships between elements of the language.

Trademarks

ArchiMate®, DirecNet®, Jericho Forum®, Making Standards Work®, OpenPegasus®, The Open Group®, TOGAF®, and UNIX® are registered trademarks and Boundaryless Information Flow™, Dependability Through Assuredness™, FACE™, Open Platform 3.0™, and The Open Group Certification Mark™ are trademarks of The Open Group.

Java® is a registered trademark of Oracle and/or its affiliates.

MDA®, Model Driven Architecture®, OMG®, and UML® are registered trademarks and BPMN™, Business Process Modeling Notation™, MOF™, and Unified Modeling Language™ are trademarks of the Object Management Group.

All other brands, company, and product names are used for identification purposes only and may be trademarks that are the sole property of their respective owners.

Acknowledgements

The Open Group gratefully acknowledges The Open Group ArchiMate Forum for developing this and earlier versions of this Open Group Standard.

The Open Group gratefully acknowledges the contribution of the following people in the development of this and earlier versions of this Open Group Standard:

- Maria-Eugenia Iacob, University of Twente
- Henk Jonkers, BiZZdesign BV
- Marc M. Lankhorst, formerly Novay, now BiZZdesign BV
- Erik (H.A.) Proper, Public Research Centre Henri Tudor & Radboud University Nijmegen
- Dick A.C. Quartel, BiZZdesign BV

The Open Group and ArchiMate project team would like to thank in particular the following individuals for their support and review of this and earlier versions of this Open Group Standard:

- Iver Band
- Mary Beijleveld
- Alexander Bielowski
- Adrian Campbell
- John Coleshaw
- Jörgen Dahlberg
- Garry Doherty
- Wilco Engelsman
- Roland Ettema
- Henry M. Franken
- Sonia González
- Kirk Hansen
- Jos van Hillegersberg
- Andrew Josey
- Louw Labuschagne
- Veer Muchandi
- Michelle Nieuwoudt
- Bill Poole
- Henk Volbeda
- Egon Willemsz
- Raina Wissing

The results presented in this Open Group Standard have largely been produced during the ArchiMate project, and The Open Group gratefully acknowledges the contribution of the many people – former members of the project team – who have contributed to them.

The ArchiMate project comprised the following organizations:

- ABN AMRO
- Centrum voor Wiskunde en Informatica
- Dutch Tax and Customs Administration
- Leiden Institute of Advanced Computer Science
- Novay
- Ordina
- Radboud Universiteit Nijmegen
- Stichting Pensioenfonds ABP

Referenced Documents

The following documents are referenced in this Open Group Standard:

[1] ISO/IEC 42010:2007, Systems and Software Engineering – Recommended Practice for Architectural Description of Software-Intensive Systems, Edition 1.

[2] Enterprise Architecture at Work: Modeling, Communication, and Analysis, M.M. Lankhorst et al, Springer, 2005.

[3] Architecture Principles: The Cornerstones of Enterprise Architecture, D. Greefhorst, E. Proper, Springer, 2011.

[4] TOGAF, Version 9.1, The Open Group, 2011.

[5] A Framework for Information Systems Architecture, J.A. Zachman, IBM Systems Journal, Volume 26, No. 3, pp. 276–292, 1987.

[6] ITU Recommendation X.901 | ISO/IEC 10746-1:1998, Information Technology – Open Distributed Processing – Reference Model – Part 1: Overview, International Telecommunication Union, 1996.

[7] Unified Modeling Language: Infrastructure, Version 2.0 (formal/05-05-05), Object Management Group, March 2006.

[8] Extending and Formalizing the Framework for Information Systems Architecture, J.F. Sowa, J.A. Zachman, IBM Systems Journal, Volume 31, No. 3, pp. 590-616, 1992.

[9] Enterprise Ontology: Theory and Methodology, J.L.G. Dietz, Springer, 2006.

[10] Unified Modeling Language: Superstructure, Version 2.0 (formal/05-07-04), Object Management Group, August 2005.

[11] A Business Process Design Language, H. Eertink, W. Janssen, P. Oude Luttighuis, W. Teeuw, C. Vissers, in Proceedings of the First World Congress on Formal Methods, Toulouse, France, September 1999.

[12] Enterprise Business Architecture: The Formal Link between Strategy and Results, R. Whittle, C.B. Myrick, CRC Press, 2004.

[13] Composition of Relations in Enterprise Architecture, R.v. Buuren, H. Jonkers, M.E. Iacob, P. Strating, in Proceedings of the Second International Conference on Graph Transformation, pp. 39–53, Edited by H. Ehrig et al, Rome, Italy, 2004.

[14] Viewpoints: A Framework for Integrating Multiple Perspectives in System Development, A. Finkelstein, J. Kramer, B. Nuseibeh, L. Finkelstein, M. Goedicke, in International Journal on Software

Engineering and Knowledge Engineering, Volume 2, No. 1, pp. 31–58, 1992.

[15] Viewpoints for Requirements Definition, G. Kotonya, I. Sommerville, IEE/BCS Software Engineering Journal, Volume 7, No. 6, pp. 375–387, November 1992.

[16] Paradigm Shift – The New Promise of Information Technology, D. Tapscott, A. Caston, New York: McGraw-Hill, 1993.

[17] The 4+1 View Model of Architecture, P.B. Kruchten, IEEE Software, Volume 12, No. 6, pp. 42–50, 1995.

[18] Model-Driven Architecture: Applying MDA to Enterprise Computing, D. Frankel, Wiley, 2003.

[19] Performance and Cost Analysis of Service-Oriented Enterprise Architectures, H. Jonkers, M. E. Iacob, in Global Implications of Modern Enterprise Information Systems: Technologies and Applications, Edited by A. Gunasekaran, IGI Global, 2009.

[20] Business Process Modeling Notation Specification (dtc/06-02-01), Object Management Group, February 2006.

[21] The Chaos Report, The Standish Group, 1994.

[22] No Silver Bullet: Essence and Accidents of Software Engineering, F.P. Brooks, IEEE Computer, 20(4):10–19, 1987.

[23] Managing Successful Programs, Office of Government Commerce (OGC), Stationery Office Books, 2007.

[24] Managing Successful Projects with PRINCE2 – 2009 Edition, Office of Government Commerce (OGC), Stationery Office Books, 2009.

[25] A Guide to the Project Management Body of Knowledge (PMBoK Guide), Fifth Edition, Project Management Institute, 2013.

Chapter 1

Introduction

An architecture is typically developed because key people have concerns that need to be addressed by the business and IT systems within the organization. Such people are commonly referred to as the "stakeholders" in the system. The role of the architect is to address these concerns, by identifying and refining the requirements that the stakeholders have, developing views of the architecture that show how the concerns and the requirements are going to be addressed, and by showing the trade-offs that are going to be made in reconciling the potentially conflicting concerns of different stakeholders. Without the architecture, it is unlikely that all the concerns and requirements will be considered and met.

Architecture descriptions are formal descriptions of a system, organized in a way that supports reasoning about the structural and behavioral properties of the system and its evolution. They define the components or building blocks that make up the overall system, and provide a plan from which products can be procured, and subsystems developed, that will work together to implement the overall system. It thus enables you to manage your overall IT investment in a way that meets the needs of your business.

To provide a uniform representation for diagrams that describe enterprise architectures, the ArchiMate enterprise architecture modeling language has been developed. It offers an integrated architectural approach that describes and visualizes the different architecture domains and their underlying relations and dependencies.

ArchiMate is a lightweight and scalable language in several respects:

- Its architecture framework is simple but comprehensive enough to provide a good structuring mechanism for architecture domains, layers, and aspects.
- The language incorporates the concepts of the "service orientation" paradigm that promotes a new organizing principle in terms of (business, application, and infrastructure) services for organizations, with far-reaching consequences for their enterprise architecture.

The role of the ArchiMate standard is to provide a graphical language for the representation of enterprise architectures over time (i.e., including transformation and migration planning), as well as their motivation and rationale. The evolution of the standard is closely linked to the developments of the TOGAF standard and the emerging results from The Open Group forums and work groups active in this area. As a consequence, the ArchiMate standard does not provide its own set of defined terms, but rather follows those provided by the TOGAF standard.

This specification contains the formal definition of ArchiMate as a visual design language with adequate concepts for specifying inter-related architectures, and specific viewpoints for selected stakeholders. This is complemented by some considerations regarding language extension mechanisms, analysis, and methodological support. Furthermore, this document is accompanied by a separate document, in which certification and governance procedures surrounding the specification are specified.

Chapter 2

Language Structure

The unambiguous specification and description of enterprise architecture's components and especially of their relationships requires an architecture modeling language that addresses the issue of consistent alignment and facilitates a coherent modeling of enterprise architectures.

This chapter presents the construction of the ArchiMate architecture modeling language. The precise definition and illustration of its generic set of core concepts and relationships follow in Chapters 3, 4, 5, 6 and 7. The concepts and relationships of the two language extensions are described in more detail in Chapters 10 and 11. They provide a proper basis for visualization, analysis, tooling, and use of these concepts and relationships.

Sections 2.1 through 2.5 discuss some general ideas, principles, and assumptions underlying the development of the ArchiMate metamodel. Section 2.6 presents the ArchiMate Framework, which is used in the remainder of this document as a reference taxonomy scheme for architecture concepts, models, viewpoints, and views. Sections 2.7 and 2.8 describe the basic structure of the two language extensions. Section 2.9 briefly describes the relationship between ArchiMate and TOGAF.

2.1 Design Approach

A key challenge in the development of a general metamodel for enterprise architecture is to strike a balance between the specificity of languages for individual architecture domains, and a very general set of architecture concepts, which reflects a view of systems as a mere set of inter-related entities. Figure 1 illustrates that concepts can be described at different levels of specialization.

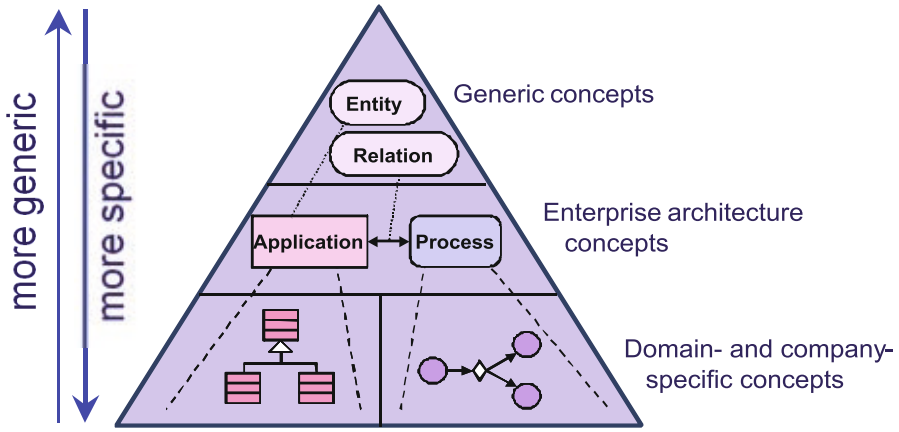

Figure 1: Metamodels at Different Levels of Specificity

At the base of the triangle we find the metamodels of the architecture modeling concepts used by specific organizations, as well as a variety of existing modeling languages and standards; UML is an example of a language in this category. At the top of the triangle we find the "most general" metamodel for system architectures, essentially a metamodel that merely comprises notions such as "entity" and "relation".

The design of the ArchiMate language started from a set of relatively generic concepts (higher up in the pyramid). These have been specialized towards application at different architectural layers, as explained below in the following sections.

The most important design restriction on the language is that it has been explicitly designed to be as small as possible, but still usable for most enterprise architecture modeling tasks. Many other languages, such as UML 2.0, try to accommodate all needs of all possible users. In the interest of simplicity of learning and use, ArchiMate has been limited to the concepts that suffice for modeling the proverbial 80% of practical cases.

2.2 Core Concepts

The core language consists of three main types of elements (note, however, that the model elements often represent *classes* of entities in the real world): *active structure* elements, *behavior* elements, and *passive structure* elements

(*objects*). The active structure elements are the business actors, application components, and devices that display actual behavior; i.e., the 'subjects' of activity (right side of the Figure 2).

> An active structure element is defined as an entity that is capable of performing behavior.

Then there is the behavioral or dynamic aspect (center of Figure 2). The active structure concepts are assigned to behavioral concepts, to show who or what performs the behavior.

> A behavior element is defined as a unit of activity performed by one or more active structure elements.

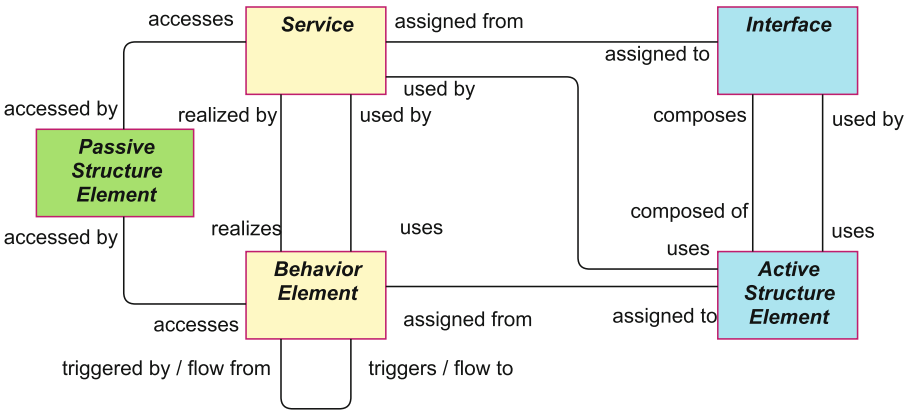

Figure 2: Generic Metamodel: The Core Concepts of ArchiMate[1]

The passive structure elements are the objects on which behavior is performed.

1 In this figure, and all the other metamodel pictures in this document, a convention for role names of relationships is used that is similar to UML (but using verbs instead of nouns). For example, a Behavior Element *realizes* a Service, and a Service *is realized by* a Behavior Element. If no cardinality is shown for a relationship end, a default of 0..* (zero or more) is assumed; if the default does not apply, the cardinality is shown explicitly in the metamodel.