The Open Group Standard

TOGAF® Standard — Architecture Content

The Open Group

The Open Group Standard

TOGAF® Standard — Architecture Content

ISBN: 1-947754-90-4
Document Number: C220

Published by The Open Group, 2005-2022.

Any comments relating to the material contained in this document may be submitted by email to:

ogspecs@opengroup.org

Contents

List of Figures

Preface

The Open Group

The Open Group is a global consortium that enables the achievement of business objectives through technology standards. With more than 870 member organizations, we have a diverse membership that spans all sectors of the technology community — customers, systems and solutions suppliers, tool vendors, integrators and consultants, as well as academics and researchers.

The mission of The Open Group is to drive the creation of Boundaryless Information Flow™ achieved by:

- Working with customers to capture, understand, and address current and emerging requirements, establish policies, and share best practices
- Working with suppliers, consortia, and standards bodies to develop consensus and facilitate interoperability, to evolve and integrate specifications and open source technologies
- Offering a comprehensive set of services to enhance the operational efficiency of consortia
- Developing and operating the industry's premier certification service and encouraging procurement of certified products

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 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/library.

The TOGAF® Standard

The TOGAF Standard is an open, industry consensus framework for Enterprise Architecture.

It is a foundational framework, which means that it is applicable to the development of any kind of architecture in any context. This foundational framework is supplemented by The Open Group TOGAF Library,[1] an extensive and growing portfolio of guidance material, providing practical guidance in the application of the TOGAF framework in specific contexts.

The TOGAF Documentation

The TOGAF documentation consists of a set of documents:

- The TOGAF Standard, which describes the generally applicable approach to Enterprise and IT Architecture
- The TOGAF Library, a portfolio of additional guidance material, which supports the practical application of the TOGAF approach

1. The TOGAF Library (see www.opengroup.org/togaf-library) is a structured library of resources that support the TOGAF Standard.

This Document

This is the TOGAF Standard — Architecture Content.

Intended Audience

The TOGAF Standard is intended for Enterprise Architects, Business Architects, IT Architects, Data Architects, Systems Architects, Solution Architects, and anyone responsible for the architecture function within an organization.

Acknowledgements

The Open Group is grateful for the contribution of many individuals and organizations in the development of the TOGAF Standard. See the TOGAF Standard — Introduction and Core Concepts for details.

Figure 3-1 is reprinted and adapted from Figure 2 of ISO/IEC/IEEE 42010:2011, Systems and Software Engineering — Architecture Description, with permission from IEEE®. Copyright© 2011, by IEEE. The IEEE disclaims any responsibility or liability resulting from the placement and use in the described manner.

Trademarks

ArchiMate, DirecNet, Making Standards Work, Open O logo, Open O and Check Certification logo, The Open Group, TOGAF, UNIX, UNIXWARE, and the Open Brand X logo are registered trademarks and Boundaryless Information Flow, Build with Integrity Buy with Confidence, Commercial Aviation Reference Architecture, Dependability Through Assuredness, Digital Practitioner Body of Knowledge, DPBoK, EMMM, FACE, the FACE logo, FHIM Profile Builder, the FHIM logo, FPB, Future Airborne Capability Environment, IT4IT, the IT4IT logo, O-AA, O-DEF, O-HERA, O-PAS, Open Agile Architecture, Open FAIR, Open Footprint, Open Process Automation, Open Subsurface Data Universe, Open Trusted Technology Provider, OSDU, Sensor Integration Simplified, SOSA, and the SOSA logo are trademarks of The Open Group.

CMMI is a registered trademark of CMMI Institute.

Energistics is a registered trademark of Energistics in the United States.

IEEE is a registered trademark of the Institute of Electrical and Electronics Engineers, Inc.

ITIL, MSP, and PRINCE2 are registered trademarks of AXELOS Limited.

Object Management Group, OMG, and UML are registered trademarks and BPMN, Business Process Modeling Notation, and Unified Modeling Language are trademarks of the Object Management Group.

PMBOK is a registered trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.

Zachman is a registered trademark of Zachman International, Inc.

The Open Group acknowledges that there may be other company names and products that might be covered by trademark protection and advises the reader to verify them independently.

Referenced Documents

Please refer to the TOGAF Standard — Introduction and Core Concepts: Appendix A for documents referenced in the TOGAF Standard.

Chapter 1: Introduction

This chapter provides an introduction to the guidance provided in the TOGAF Standard — Architecture Content (this document).

1.1 Overview

Architects executing the Architecture Development Method (ADM) will produce a number of outputs as a result of their efforts, such as process flows, architectural requirements, project plans, or project compliance assessments. The Content Framework provides a structural model for architectural content that allows the major work products that an architect creates to be consistently defined, structured, and presented.

The Content Framework provided here is intended to allow the TOGAF framework to be used as a stand-alone framework for architecture within an enterprise. However, other Content Frameworks exist (such as the Zachman® Framework) and it is anticipated that some enterprises may opt to use an external framework in conjunction with the TOGAF framework. In these cases, the TOGAF Content Framework provides a useful reference and starting point for TOGAF content to be mapped to other Content Frameworks.

The Architecture Content Framework uses the following three categories to describe the type of architectural work product within the context of use:

- A **deliverable** is a work product that is contractually specified and in turn formally reviewed, approved, and signed off by the stakeholders

  Deliverables represent the output of projects and those deliverables that are in documentation form will typically be archived at completion of a project, or transitioned into an Architecture Repository as a reference model, standard, or snapshot of the Architecture Landscape at a point in time.

- An **artifact** is an architectural work product that describes an aspect of the architecture

  Artifacts are generally classified as catalogs (lists of things), matrices (showing relationships between things), and diagrams (pictures of things). Examples include a requirements catalog, application interaction matrix, and a value chain diagram. An architectural deliverable may contain many artifacts and artifacts will form the content of the Architecture Repository.

- A **building block** represents a potentially re-usable component of enterprise capability that can be combined with other building blocks to deliver architectures and solutions

  Building blocks can be defined at various levels of detail, depending on what stage of architecture development has been reached. For instance, at an early stage, a building block can simply consist of a name or an outline description. Later on, a building block may be decomposed into multiple supporting building blocks and may be accompanied by a full specification. Building blocks can relate to "architectures" or "solutions".

— **Architecture Building Blocks (ABBs)** typically describe what is required of SBBs at a more logical or supplier-independent level; those requirements may include services to be performed, data resources, and capabilities needed. ABBs include logical business, application, and technology components

— **Solution Building Blocks (SBBs)** represent physical or supplier-specific components that have the capability to realize part or all of a more logical ABB. There are business, application, and technology SBBs.

The relationships between deliverables, artifacts, and building blocks are shown in Figure 1-1.

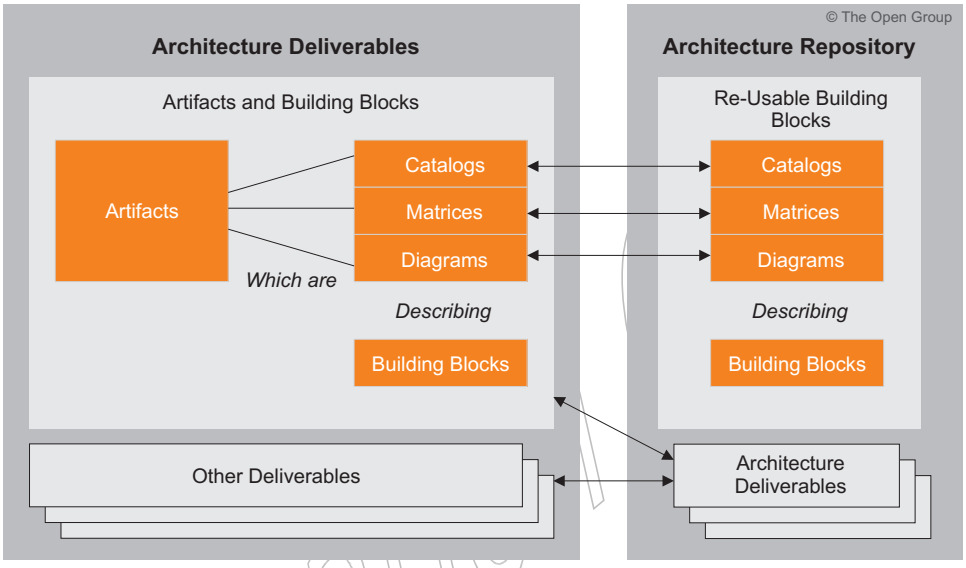

Figure 1-1 Relationships between Deliverables, Artifacts, and Building Blocks

For example, an Architecture Definition Document is a deliverable that documents an Architecture Description. This document will contain a number of complementary artifacts that are architecture views of the building blocks relevant to the architecture. For example, a process flow diagram (an artifact) may be created to describe the target call handling process (a building block). This artifact may also describe other building blocks, such as the actors involved in the process (e.g., a Customer Services Representative). An example of the relationships between deliverables, artifacts, and building blocks is illustrated in Figure 1-2.

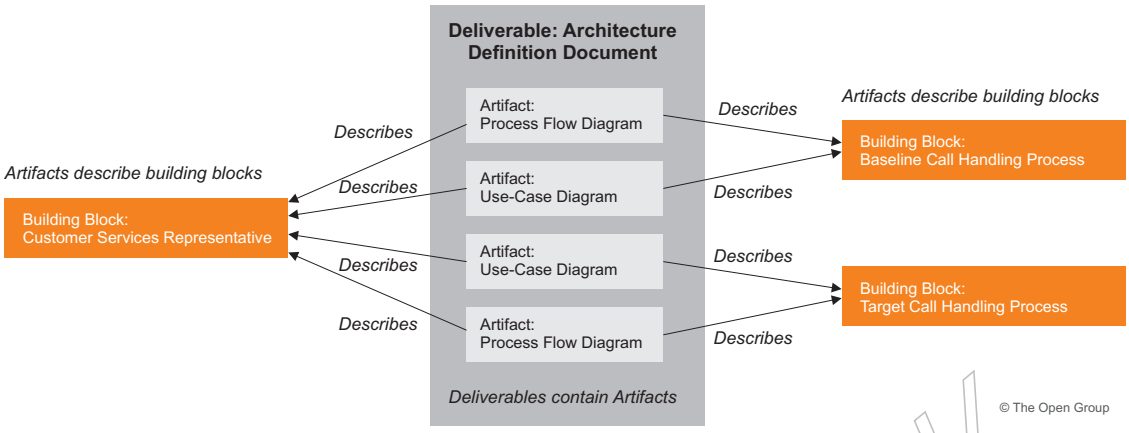

Figure 1-2 Example — Architecture Definition Document

1.2 TOGAF Content Framework and Enterprise Metamodel

1.2.1 Overview

The TOGAF ADM provides lifecycle management to create and manage architectures within an enterprise. At each phase within the ADM, a discussion of inputs, outputs, and steps describes a number of architecture work products.

An essential task when establishing the enterprise-specific Enterprise Architecture Capability in the Preliminary Phase of the ADM is to define:

- A categorization framework to be used to structure the Architecture Descriptions, the work products used to express an architecture, and the collection of models that describe the architecture; this is referred to as the **Content Framework**
- An understanding of the types of entities within the enterprise and the relationships between them that need to be captured, stored, and analyzed in order to create the Architecture Description; this **Enterprise Metamodel** depicts this information in the form of a formal model
- The specific artifacts to be developed (see Chapter 4)

The Content Framework chosen is likely to be influenced by:

- The Architecture Framework selected as the basis for the Enterprise Architecture Capability
- The chosen software tool used to support the Enterprise Architecture Capability

1.2.2 Content Framework

The Content Framework defines a categorization framework to be used to structure the Architecture Description, the work product used to express an architecture, and the collection of models that describe the architecture.

The Architecture Repository, which is explained in Section 4.2.5, is structured to store the artifacts and work products identified in the Content Framework. The Content Framework is one element of the Enterprise-Specific Architecture Framework.

1.2.3 Enterprise Metamodel

The TOGAF Standard encourages development of an Enterprise Metamodel, which defines the types of entity to appear in the models that describe the enterprise, together with the relationships between these entities.

An Enterprise Metamodel provides value in several ways:

- It gives architects a starter set of the types of thing to investigate and to cover in their models
- It provides a form of completeness-check for any architecture modeling language, or architecture metamodel, that is proposed for use in an enterprise

  Namely, how completely does it handle the types of entity in the Enterprise Metamodel, and manage required facts about them such as their attributes and relationships?
- It can help ensure:
  - Consistency
  - Completeness
  - Traceability

Note that the TOGAF Standard does not aim to constrain an enterprise's:

- Selection of artifacts
- Modeling notation

  The TOGAF Standard may use the ArchiMate® modeling language, Business Process Modeling Notation™ (BPMN™), Unified Modeling Language™ (UML®), entity relationship diagramming, flowcharting, or any other notation that can express some TOGAF ideas.

The types of entity within an enterprise and the relationships between them are specific to the individual enterprise. Developing a high-quality metamodel is an important aspect of establishing the Enterprise Architecture Capability.

1.2.4 The TOGAF Content Framework

The TOGAF Content Framework defines a categorization framework to be used to structure the Architecture Description, the work products used to express an architecture, and the collection of models that describe the architecture.

There are many alternative Content Frameworks (e.g., the TOGAF Content Framework, the Zachman Framework, DoDAF, NAF, etc.). Selecting a Content Framework is essential even though the choice of Content Framework is less important. The final Content Framework is usually adapted to fit specific organization needs.

The TOGAF Content Framework is intended to:

- Provide a detailed model of architectural work products
- Drive consistency in the outputs created when following the ADM
- Provide a comprehensive checklist of architecture output that could be created
- Reduce the risk of gaps within the final architecture deliverable set
- Help an enterprise mandate standard architecture concepts, terms, and deliverables

At the highest level, the TOGAF Content Framework (see Figure 1-3) is structured in line with the phases of the ADM.

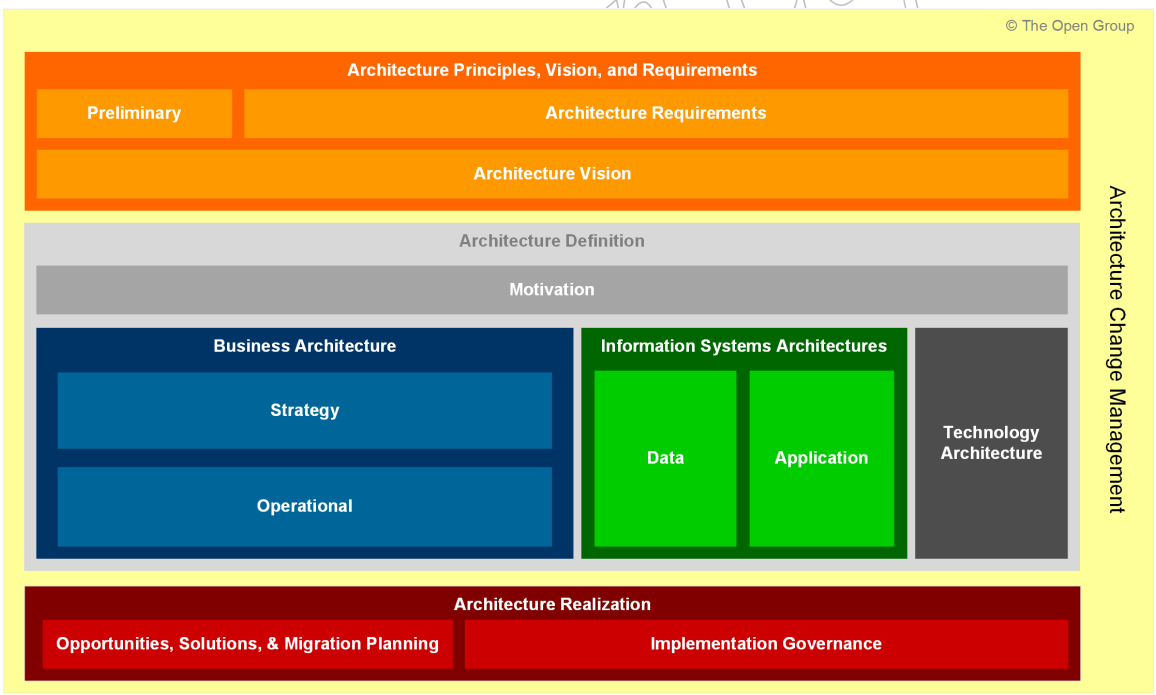

Figure 1-3 Content Framework by ADM Phase

- **Architecture Principles, Vision, Motivation, and Requirements** models are intended to capture the surrounding context of formal architecture models, including general Architecture Principles, strategic context that forms input for architecture modeling, and requirements generated from the architecture

  The relevant aspects of the business context that have given rise to the Request for Architecture work are typically investigated, refined, validated, and recorded in the Preliminary and Architecture Vision phases.

- **Business Architecture** captures architecture models of the business, looking specifically at factors that motivate the enterprise, its structure, and its capabilities
- **Information Systems Architecture** models capture architecture models of IT systems, looking at applications and data in line with the TOGAF ADM phases
- **Technology Architecture** models capture technology assets that are used to implement and realize information system solutions
- **Architecture Realization/Transformation** models capture change roadmaps showing transition between architecture states and binding statements that are used to steer and govern an implementation of the architecture
- **Architecture Change Management** models capture value realization management events, internal and external, that impact the Enterprise Architecture and the generation of requirements for action

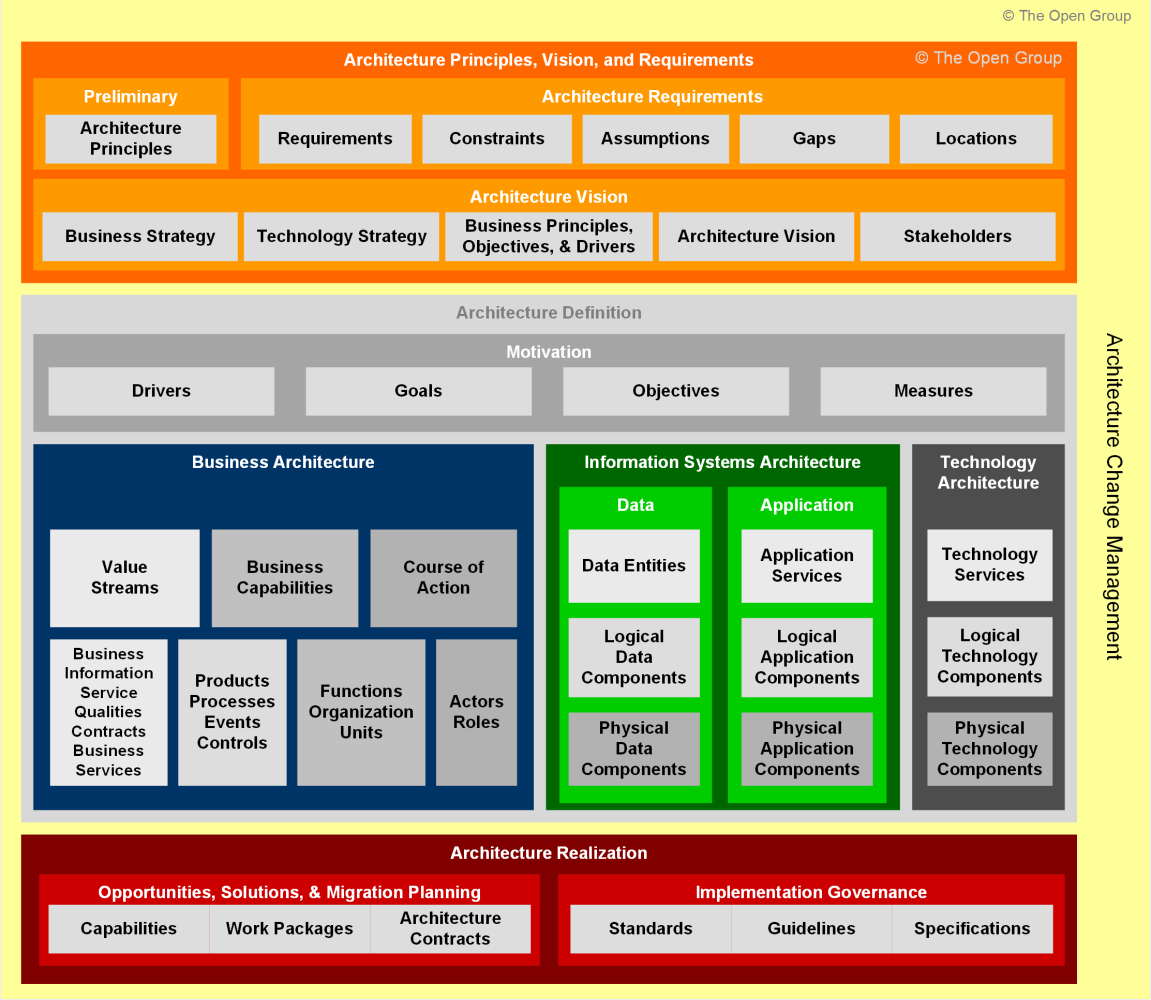

Figure 1-4 Content Framework Overview

1.3 Content Framework and the TOGAF ADM

The TOGAF ADM describes the process of moving from a baseline state of the enterprise to a target state of the enterprise. The ADM will address a business need through a process of visioning, architecture definition, transformation planning, and Architecture Governance. At each stage in this process, the ADM requires information as inputs and will create outputs as a result of executing a number of steps. The Content Framework provides an underlying structure for the ADM that defines inputs and outputs in more detail and puts each deliverable into the context of the holistic architecture view of the enterprise.

The Content Framework should therefore be used as a companion to the ADM. The ADM describes what needs to be done to create an architecture and the Content Framework describes what the architecture should look like once it is done.

1.4 The Enterprise Continuum

It is usually impossible to create a single unified architecture that meets all the requirements of all stakeholders for all time. Therefore, the Enterprise Architect will need to deal not just with a single Enterprise Architecture, but with many related Enterprise Architectures.

Each architecture may have a different purpose and architectures may relate to one another. Effectively bounding the scope of an architecture is therefore a Critical Success Factor (CSF) in allowing architects to break down a complex problem space into manageable components that can be individually addressed.

The Enterprise Continuum provides a view of the Architecture Repository that shows the evolution of these related architectures from generic to specific, from abstract to concrete, and from logical to physical.

Chapter 6 discusses the Enterprise Continuum; including the Architecture Continuum and the Solutions Continuum.

1.5 The Architecture Repository

Operating a mature Architecture Capability within a large enterprise creates a huge volume of architectural output. Effective management and leverage of these architectural work products require a formal taxonomy for different types of architectural asset alongside dedicated processes and tools for architectural content storage.

Chapter 7 provides a structural framework for an Architecture Repository that allows an enterprise to distinguish between different types of architectural assets that exist at different levels of abstraction in the organization.

Chapter 2: TOGAF Content Framework and Enterprise Metamodel

2.1 Overview

The TOGAF ADM provides a process lifecycle to create and manage architectures within an enterprise. At each phase within the ADM, a discussion of inputs, outputs, and steps describes a number of architectural work products or artifacts, such as process and application.

The Content Framework and Enterprise Metamodel provided here define a formal structure for these terms to ensure consistency within the ADM and also to provide guidance for organizations that wish to implement their architecture within an architecture tool.

The **Content Framework** defines a categorization framework to be used to structure the Architecture Description, the work product used to express an architecture, and the collection of models that describe the architecture.

The **Enterprise Metamodel** defines the types of entities to appear in the models that describe the enterprise, together with the relationships between these entities.

2.2 TOGAF Enterprise Metamodel Vision

The TOGAF Standard includes the TOGAF Enterprise Metamodel which captures the entities and relationships that are likely to be encountered in the majority of enterprises. This may be used as the basis for developing an Organization-Specific Metamodel when establishing the Enterprise Architecture Capability in the Preliminary Phase and also provides the context for the specific artifacts referenced in the descriptions of the ADM phases and described in detail in Chapter 3.

When developing an Organization-Specific Metamodel, architects may choose not to include entities and relationships from the TOGAF Enterprise Metamodel which are not relevant and/or add additional entities and relationships.

This section provides an overview of the TOGAF Enterprise Metamodel. Subsequent sections discuss each area of the metamodel in more detail.

2.2.1 Overview of the TOGAF Enterprise Metamodel

The TOGAF Enterprise Metamodel includes a set of entities, defined in Section 2.4, that allow architectural concepts to be captured, stored, filtered, queried, and represented in a way that supports consistency, completeness, and traceability.

The categorization mechanism of the Content Framework may be used to structure a representation of the TOGAF Enterprise Metamodel, as shown in Figure 2-1.

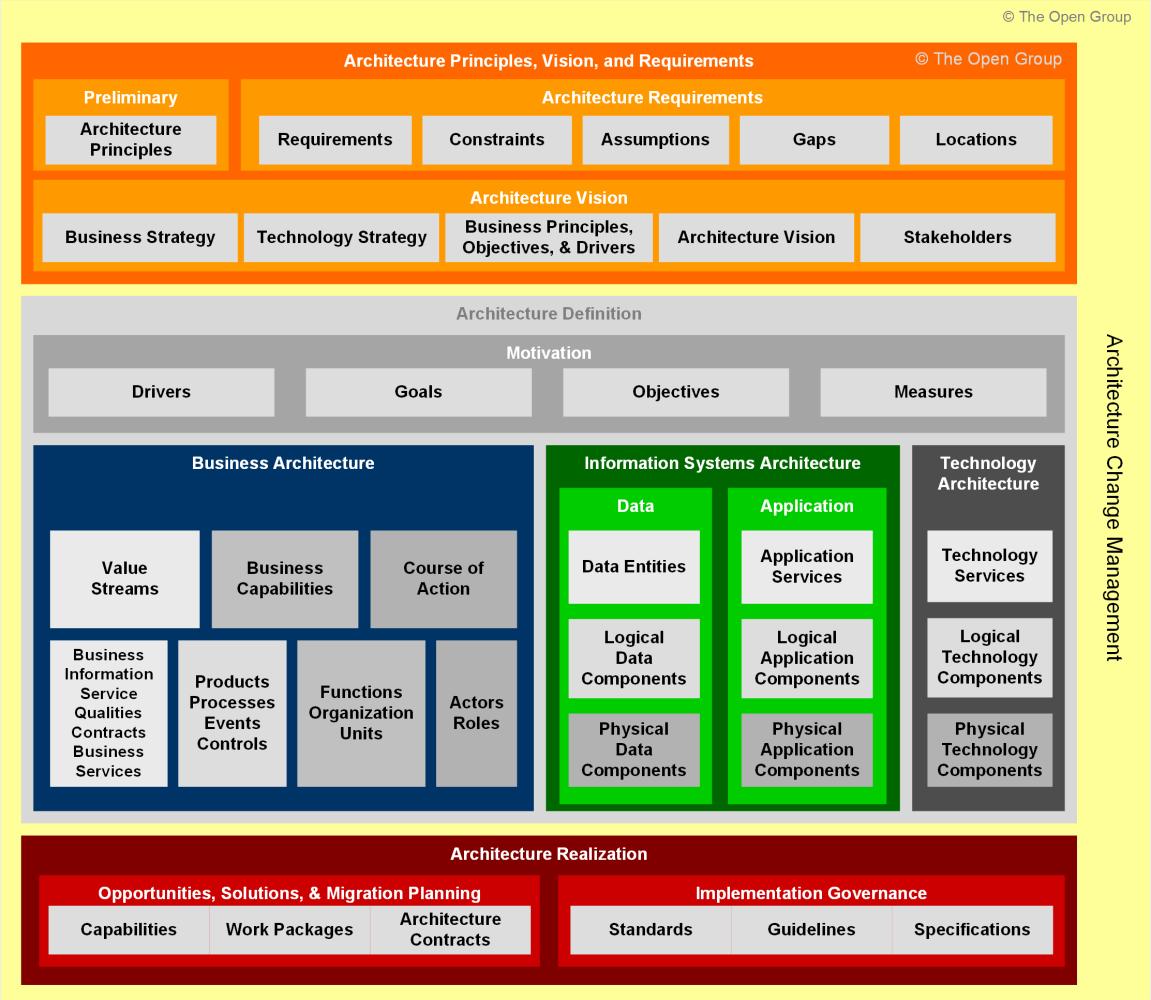

Figure 2-1 Using the Content Framework to Structure the TOGAF Enterprise Metamodel

2.3 TOGAF Enterprise Metamodel in Detail

The relationships between entities in the TOGAF Enterprise Metamodel are shown in Figure 2-2.

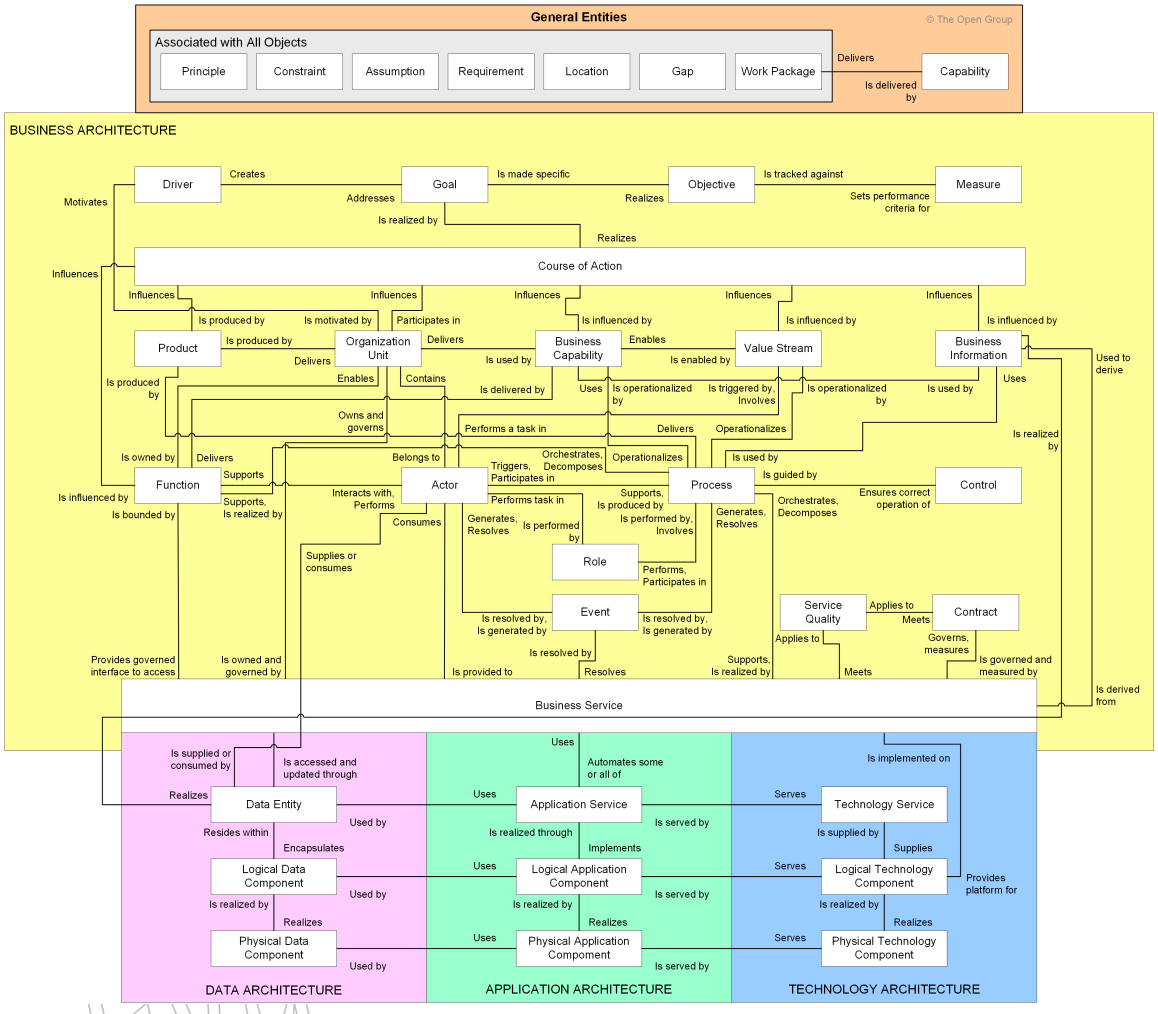

Figure 2-2 Relationships between Entities in the TOGAF Enterprise Metamodel

2.4 TOGAF Enterprise Metamodel Entities

The following table lists and describes the entities within the Enterprise Metamodel.

Metamodel Entity	Description
Actor	A person, organization, or system that has a role that initiates or interacts with activities; for example, a sales representative who travels to visit customers. Actors may be internal or external to an organization. In the automotive industry, an original equipment manufacturer would be considered an actor by an automotive dealership that interacts with its supply chain activities.
Application Service	The automated elements of a business service. An application service may deliver or support part or all of one or more business services.
Assumption	A statement of probable fact that has not been fully validated at this stage, due to external constraints. For example, it may be assumed that an existing application will support a certain set of functional requirements, although those requirements may not yet have been individually validated.
Business Capability	A particular ability that a business may possess or exchange to achieve a particular purpose.
Business Information	Represents a concept and its semantics used within the business.
Business Service	Supports the business by encapsulating a unique element of business behavior; a service offered external to the enterprise may be supported by business services.
Capability	An ability that an organization, person, or system possesses. **Note:** This a general-purpose definition. See *Business Capability* for how this concept is refined for usage in Business Architecture.
Constraint	An external factor that prevents an organization from pursuing particular approaches to meet its goals. For example, customer data is not harmonized within the organization, regionally or nationally, constraining the organization's ability to offer effective customer service.
Contract	An agreement between a consumer and a provider that establishes functional and non-functional parameters for interaction. This applies to all types of service interactions within the metamodel.
Control	A decision-making step with accompanying decision logic used to determine execution approach for a process or to ensure that a process complies with governance criteria. For example, a sign-off control on the purchase request processing process that checks whether the total value of the request is within the sign-off limits of the requester, or whether it needs escalating to higher authority.
Course of Action	Direction and focus provided by strategic goals and objectives, often to deliver the value proposition characterized in the business model.
Data Entity	Represents data that is recognized by the business as a distinct concept.

Metamodel Entity	Description
Driver	An external or internal condition that motivates the organization to define its goals. An example of an external driver is a change in regulation or compliance rules which, for example, require changes to the way an organization operates; i.e., Sarbanes-Oxley in the US.
Event	An organizational state change that triggers processing events; may originate from inside or outside the organization and may be resolved inside or outside the organization.
Function	A set of business behaviors based on a chosen set of criteria. Functions are usually close-coupled to/with organizational units.
Gap	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. **Note:** Gap analysis is described in the TOGAF Standard — ADM Techniques.
Goal	A high-level statement of intent or direction for an organization. Typically used to measure success of an organization.
Location	A place where activities occur. Locations can be composed and decomposed.
Logical Application Component	An encapsulation of application functionality that is definable by services offered and data maintained, independently of implementation and technology.
Logical Data Component	A data structure composed of logically-related data entities.
Logical Technology Component	An implementation-independent encapsulation of technology services.
Measure	An indicator or factor that can be tracked, usually on an ongoing basis, to determine success or alignment with objectives and goals.
Objective	An organizational aim that is declared in a Simple, Measurable, Actionable, Realistic, and Timebound (SMART) way. For example, "Increase capacity utilization by 30% by the end of the year, to support the planned increase in market share".
Organization Unit	A self-contained unit of resources with goals, objectives, and measures. Organization units may include external parties and business partner organizations.
Physical Application Component	A realization of logical application functionality using components of functionality in applications that may be hired, procured, or built.
Physical Data Component	A data structure that realizes related logical data components represented in the format or schema required by a particular technology.
Physical Technology Component	A realization of logical technology functionality using a particular technology product that may be deployed.

Metamodel Entity	Description
Principle	A qualitative statement of intent that should be met by the architecture. It has at least a supporting rationale and a measure of importance. **Note:** A sample set of Architecture Principles is defined in the TOGAF Standard — ADM Techniques.
Process	A process represents a sequence of activities that together achieve a specified outcome, can be decomposed into sub-processes, and can show operation of a business capability or service (at next level of detail). Processes may also be used to link organizations, business capabilities, services, and processes. A process may realize one service and/or orchestrate subordinate services.
Product	An outcome generated by the business to be offered to customers. Products include materials and/or services.
Requirement	A quantitative statement of business need that must be met by a particular architecture or work package.
Role	The usual or expected behavior of an actor, or the part somebody or something plays in a particular process or event. An actor may have a number of roles. See also *Actor*.
Service Quality	A configuration of non-functional requirements or attributes that may be assigned to a business, application, or technology service.
Technology Service	A technical capability required to provide enabling infrastructure that supports the delivery of applications.
Value Stream	A representation of an end-to-end collection of activities that create an overall result for a customer, stakeholder, or end-user.
Work Package	A set of actions identified 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.

2.5 TOGAF Enterprise Metamodel Attributes

The following table shows typical attributes for each of the metamodel entities described previously.

Metamodel Entity	Attribute	Description
All Metamodel Entities	ID	Unique identifier for the architecture entity.
	Name	Brief name of the architecture entity.
	Description	Textual description of the architecture entity.
	Category	User-definable categorization taxonomy for each metamodel entity.
	Source	Location from where the information was collected.
	Owner	Owner of the architecture entity.
Capability	Business value	Describes how this capability provides value to the enterprise.
	Increments	Lists possible maturity/quality levels for the capability.
Constraint	No additional attributes	This metamodel entity has only basic attributes.
Gap	No additional attributes	This metamodel entity has only basic attributes.
Location	Category	The following categories of Location apply: Region (applies to a grouping of countries or territory; e.g., South East Asia, UK, and Ireland), Country (applies to a single country; e.g., US), Building (applies to a site of operation; where several offices are collected in a single city, this category may represent a city), and Specific Location (applies to any specific location within a building, such as a server room). The nature of the business may introduce other Locations: Ship or Port for a ferry company, Mine for a gold company, Car for a police force, Hotel for any firm's traveling workers, and so on.

Metamodel Entity	Attribute	Description
Principle	Category	The following categories of principle apply: Guiding Principle, Business Principle, Data Principle, Application Principle, Integration Principle, Technology Principle.
	Priority	Priority of this principle relative to other principles.
	Statement of principle	Statement of what the principle is.
	Rationale	Statement of why the principle is required and the desired outcome to be reached.
	Implication	Statement of what the principle means in practical terms.
	Metric	Identifies mechanisms that will be used to measure whether the principle has been met or not.
Requirement	Statement of requirement	Statement of what the requirement is, including a definition of whether the requirement shall be met, should be met, or may be met.
	Rationale	Statement of why the requirement exists.
	Acceptance criteria	The parameters that will be fulfilled if the requirement is being met, together with the tests that will be carried out to assess the state of the parameters.
Actor	# FTEs	Estimated number of FTEs that operate as this actor.
	Actor goal	Objectives that this actor has, in general terms.
	Actor tasks	Tasks that this actor performs, in general terms.
Business Service	Standards class	Non-Standard, Proposed Standard, Provisional Standard, Standard, Phasing-Out Standard, Retired Standard.
	Standard creation date	If the business service is a standard, when the standard was created.
	Last standard review date	Last date that the standard was reviewed.

Metamodel Entity	Attribute	Description
	Next standard review date	Next date for the standard to be reviewed.
	Retire date	Date when the standard was/will be retired.
Contract	Behavior characteristics	Functional behavior to be supported within the scope of the contract.
	Service name "caller"	Consuming service.
	Service name "called"	Providing service.
	Service quality characteristics	Non-functional behavior to be supported within the scope of the contract.
	Availability characteristics	Degree to which something is available for use.
	Service times	Hours during which the service must be available.
	Manageability characteristics	Ability to gather information about the state of something and control it.
	Serviceability characteristics	Ability to identify problems and take corrective action, such as to repair or upgrade a component in a running system.
	Performance characteristics	Ability of a component to perform its tasks in an appropriate time.
	Response requirements	Response times that the service provider must meet for particular operations.
	Reliability characteristics	Resistance to failure.
	Quality of information required	Contracted requirements on accuracy and completeness of information.
	Contract control requirements	Level of governance and enforcement applied to the contractual parameters for overall service.
	Result control requirements	Measures in place to ensure that each service request meets contracted criteria.
	Recoverability characteristics	Ability to restore a system to a working state after an interruption.
	Locatability characteristics	Ability of a system to be found when needed.

Metamodel Entity	Attribute	Description
	Security characteristics	Ability of a system to prevent unauthorized access to functions and data.
	Privacy characteristics	Protection of data from unauthorized access.
	Integrity characteristics	Ability of a system to ensure that data has not been corrupted.
	Credibility characteristics	Ability of a system to ensure that the service request originates from an authorized source.
	Localization characteristics	Ability of a service to support localized variants for different consumer groups.
	Internationalization characteristics	Ability of a service to support international variations in business logic and data representation (such as character set).
	Interoperability characteristics	Ability of the service to interoperate with different technical environments, inside and outside of the organization.
	Scalability characteristics	Ability of the service to grow or shrink its performance or capacity appropriately to the demands of the environment in which it operates.
	Portability characteristics	Of data, people, applications, and components.
	Extensibility characteristics	Ability to accept new functionality.
	Capacity characteristics	Contracted capacity of the service provider to meet requests.
	Throughput	Required throughput capacity.
	Throughput period	Time period needed to deliver throughput capacity.
	Growth	Expected future growth rate of service request.
	Growth period	Time period needed to reach the expected growth rate.
	Peak profile short term	Short-term profile of peak service traffic.
	Peak profile long term	Long-term profile of peak service traffic.

Metamodel Entity	Attribute	Description
Control	No additional attributes	This metamodel entity has only basic attributes.
Driver	No additional attributes	This metamodel entity has only basic attributes.
Event	No additional attributes	This metamodel entity has only basic attributes.
Function	Standards class	Non-Standard, Proposed Standard, Provisional Standard, Standard, Phasing-Out Standard, Retired Standard.
	Standard creation date	If the product is a standard, when the standard was created.
	Last standard review date	Last date that the standard was reviewed.
	Next standard review date	Next date for the standard to be reviewed.
	Retire date	Date when the standard was/will be retired.
Goal	No additional attributes	This metamodel entity has only basic attributes.
Measure	No additional attributes	This metamodel entity has only basic attributes.
Objective	No additional attributes	This metamodel entity has only basic attributes.
Organization Unit	Headcount	Number of FTEs working within the organization.
Process	Standards class	Non-Standard, Proposed Standard, Provisional Standard, Standard, Phasing-Out Standard, Retired Standard.
	Standard creation date	If the product is a standard, when the standard was created.
	Last standard review date	Last date that the standard was reviewed.
	Next standard review date	Next date for the standard to be reviewed.
	Retire date	Date when the standard was/will be retired.
	Process criticality	Criticality of this process to business operations.
	Manual or automated	Whether this process is supported by IT or is a manual process.
	Process volumetrics	Data on frequency of process execution.

Metamodel Entity	Attribute	Description
Product	No additional attributes	This metamodel entity has only basic attributes.
Role	Estimated number of FTEs that operate in this Role	This metamodel entity has only basic attributes.
Service Quality	No additional attributes	This metamodel entity has only basic attributes.
Service	Standards class	Non-Standard, Proposed Standard, Provisional Standard, Standard, Phasing-Out Standard, Retired Standard.
	Standard creation date	If the product is a standard, when the standard was created.
	Last standard review date	Last date that the standard was reviewed.
	Next standard review date	Next date for the standard to be reviewed.
	Retire date	Date when the standard was/will be retired.
Application Component	Standards class	Non-Standard, Proposed Standard, Provisional Standard, Standard, Phasing-Out Standard, Retired Standard.
	Standard creation date	If the product is a standard, when the standard was created.
	Last standard review date	Last date that the standard was reviewed.
	Next standard review date	Next date for the standard to be reviewed.
	Retire date	Date when the standard was/will be retired.
Application Service	Standards class	Non-Standard, Proposed Standard, Provisional Standard, Standard, Phasing-Out Standard, Retired Standard.
	Standard creation date	If the product is a standard, when the standard was created.
	Last standard review date	Last date that the standard was reviewed.
	Next standard review date	Next date for the standard to be reviewed.
	Retire date	Date when the standard was/will be retired.
Logical Application Component	Standards class	Non-Standard, Proposed Standard, Provisional Standard, Standard, Phasing-Out Standard, Retired Standard.

Metamodel Entity	Attribute	Description
	Standard creation date	If the product is a standard, when the standard was created.
	Last standard review date	Last date that the standard was reviewed.
	Next standard review date	Next date for the standard to be reviewed.
	Retire date	Date when the standard was/will be retired.
Physical Application Component	Lifecycle status	Proposed, In Development, Live, Phasing Out, Retired.
	Standards class	Non-Standard, Proposed Standard, Provisional Standard, Standard, Phasing-Out Standard, Retired Standard.
	Standard creation date	If the product is a standard, when the standard was created.
	Last standard review date	Last date that the standard was reviewed.
	Next standard review date	Next date for the standard to be reviewed.
	Retire date	Date when the standard was/will be retired.
	Initial live date	Date when the first release of the application was/will be released into production.
	Date of last release	Date when the last release of the application was released into production.
	Date of next release	Date when the next release of the application will be released into production.
	Retirement date	Date when the application was/will be retired.
	Availability characteristics	Degree to which something is available for use.
	Service times	Hours during which the application must be available.
	Manageability characteristics	Ability to gather information about the state of something and control it.
	Serviceability characteristics	Ability to identify problems and take corrective action, such as to repair or upgrade a component in a running system.
	Performance characteristics	Ability of a component to perform its tasks in an appropriate time.

Metamodel Entity	Attribute	Description
	Reliability characteristics	Resistance to failure.
	Recoverability characteristics	Ability to restore a system to a working state after an interruption.
	Locatability characteristics	Ability of a system to be found when needed.
	Security characteristics	Ability of a system to prevent unauthorized access to functions and data.
	Privacy characteristics	Protection of data from unauthorized access.
	Integrity characteristics	Ability of a system to ensure that data has not been corrupted.
	Credibility characteristics	Ability of a system to ensure that the service request originates from an authorized source.
	Localization characteristics	Ability of a service to support localized variants for different consumer groups.
	Internationalization characteristics	Ability of a service to support international variations in business logic and data representation (such as character set).
	Interoperability characteristics	Ability of the service to interoperate with different technical environments, inside and outside of the organization.
	Scalability characteristics	Ability of the service to grow or shrink its performance or capacity appropriately to the demands of the environment in which it operates.
	Portability characteristics	Of data, people, applications, and components.
	Extensibility characteristics	Ability to accept new functionality.
	Capacity characteristics	Contracted capacity of the service provider to meet requests.
	Throughput	Required throughput capacity.
	Throughput period	Time period needed to deliver throughput capacity.
	Growth	Expected future growth rate of service request.

Metamodel Entity	Attribute	Description
	Growth period	Time period needed to reach the expected growth rate.
	Peak profile short term	Short-term profile of peak service traffic.
	Peak profile long term	Long-term profile of peak service traffic.
Data Entity	Category	The following categories of data entity apply: Message, Internally Stored Entity.
	Privacy classification	Level of restriction placed on access to the data.
	Retention classification	Level of retention to be placed on the data.
Logical Data Component	Standards class	Non-Standard, Proposed Standard, Provisional Standard, Standard, Phasing-Out Standard, Retired Standard.
	Standard creation date	If the product is a standard, when the standard was created.
	Last standard review date	Last date that the standard was reviewed.
	Next standard review date	Next date for the standard to be reviewed.
	Retire date	Date when the standard was/will be retired.
Physical Data Component	Standards class	Non-Standard, Proposed Standard, Provisional Standard, Standard, Phasing-Out Standard, Retired Standard.
	Standard creation date	If the product is a standard, when the standard was created.
	Last standard review date	Last date that the standard was reviewed.
	Next standard review date	Next date for the standard to be reviewed.
	Retire date	Date when the standard was/will be retired.
Logical Technology Component	Standards class	Non-Standard, Proposed Standard, Provisional Standard, Standard, Phasing-Out Standard, Retired Standard.
	Standard creation date	If the product is a standard, when the standard was created.
	Last standard review date	Last date that the standard was reviewed.
	Next standard review date	Next date for the standard to be reviewed.

Metamodel Entity	Attribute	Description
	Retire date	Date when the standard was/will be retired.
	Category	Logical Technology Components are categorized according to the defined taxonomy (such as the TOGAF Technical Reference Model (TRM)), adapted to meet the needs of an individual organization.
Physical Technology Component	Standards class	Non-Standard, Proposed Standard, Provisional Standard, Standard, Phasing-Out Standard, Retired Standard.
	Standard creation date	If the product is a standard, when the standard was created.
	Last standard review date	Last date that the standard was reviewed.
	Next standard review date	Next date for the standard to be reviewed.
	Retire date	Date when the standard was/will be retired.
	Category	Physical Technology Components are categorized according to the defined taxonomy (such as the TOGAF TRM), adapted to meet the needs of an individual organization.
	Product name	Name of the product making up the technology component.
	Module name	Module, or other sub-product, name making up the technology component.
	Vendor	Vendor providing the technology component.
	Version	Version of the product making up the technology component.
Technology Service	Standards class	Non-Standard, Proposed Standard, Provisional Standard, Standard, Phasing-Out Standard, Retired Standard.

Metamodel Entity	Attribute	Description
	Category	Technology Services are categorized according to the defined taxonomy (such as the TOGAF TRM), adapted to meet the needs of an individual organization.
Business Capability	No additional attributes	This metamodel entity has only basic attributes.
Technology Component	Standards class	Non-Standard, Proposed Standard, Provisional Standard, Standard, Phasing-Out Standard, Retired Standard.
Course of Action	No additional attributes	This metamodel entity has only basic attributes.
Value Stream	No additional attributes	This metamodel entity has only basic attributes.
Work Package	Category Capability delivered	The following categories of work package apply: Work Package, Work Stream, Project, Program, Portfolio. Describes the contribution this work package makes to capability delivery.

2.6 TOGAF Enterprise Metamodel Relationships

Source Entity	Target Entity	Name
Actor	Actor	Decomposes
Actor	Business Service	Consumes
Actor	Data Entity	Supplies or consumes
Actor	Event	Generates
Actor	Event	Resolves
Actor	Function	Interacts with
Actor	Function	Performs
Actor	Organization Unit	Belongs to
Actor	Process	Participates in
Actor	Process	Triggers
Actor	Role	Performs task in
Actor	Value Stream	Performs a task in
Application Service	Business Service	Automates some or all of
Application Service	Data Entity	Used by
Application Service	Logical Application Component	Is realized through

Source Entity	Target Entity	Name
Application Service	Technology Service	Is served by
Business Capability	Business Information	Uses
Business Capability	Course of Action	Is influenced by
Business Capability	Function	Is delivered by
Business Capability	Organization Unit	Is used by
Business Capability	Process	Is operationalized by
Business Capability	Value Stream	Enables
Business Information	Business Capability	Is used by
Business Information	Business Service	Used to derive
Business Information	Course of Action	Is influenced by
Business Information	Data Entity	Is realized by
Business Information	Process	Uses
Business Service	Actor	Is provided to
Business Service	Application Service	Uses
Business Service	Business Information	Is derived from
Business Service	Business Service Quality	Meets
Business Service	Business Service	Consumes
Business Service	Business Service	Decomposes
Business Service	Contract	Is governed and measured by
Business Service	Data Entity	Is accessed and updated through
Business Service	Event	Resolves
Business Service	Function	Provides governed interface to access
Business Service	Logical Technology Component	Is implemented on
Business Service	Organization Unit	Is owned and governed by
Business Service	Process	Is realized by
Business Service	Process	Supports
Capability	Work Package	Is delivered by
Contract	Business Service	Governs, measures
Contract	Service Quality	Meets
Control	Process	Ensures correct operation of
Course of Action	Business Capability	Influences
Course of Action	Business Information	Influences
Course of Action	Function	Influences
Course of Action	Goal	Realizes
Course of Action	Organization Unit	Influences
Course of Action	Product	Influences
Course of Action	Value Stream	Influences
Data Entity	Actor	Is supplied or consumed by
Data Entity	Application Service	Used by
Data Entity	Business Information	Realizes
Data Entity	Business Service	Is accessed and updated through
Data Entity	Data Entity	Decomposes
Data Entity	Data Entity	Relates to
Data Entity	Logical Data Component	Resides within

Source Entity	Target Entity	Name
Driver	Driver	Decomposes
Driver	Goal	Creates
Driver	Organization Unit	Motivates
Event	Actor	Is generated by
Event	Actor	Is resolved by
Event	Business Service	Is resolved by
Event	Process	Is generated by
Event	Process	Is resolved by
Function	Actor	Supports
Function	Business Capability	Delivers
Function	Business Service	Is bounded by
Function	Course of Action	Is influenced by
Function	Function	Communicates with
Function	Function	Decomposes
Function	Organization Unit	Is owned by
Function	Process	Orchestrates
Function	Process	Decomposes
Goal	Course of Action	Is realized by
Goal	Driver	Addresses
Goal	Goal	Decomposes
Goal	Objective	Is made specific
Logical Application Component	Application Service	Implements
Logical Application Component	Logical Application Component	Decomposes
Logical Application Component	Logical Application Component	Communicates with
Logical Application Component	Logical Data Component	Used by
Logical Application Component	Logical Technology Component	Is served by
Logical Application Component	Physical Application Component	Is realized by
Logical Data Component	Data Entity	Encapsulates
Logical Data Component	Logical Application Component	Uses
Logical Data Component	Physical Data Component	Is realized by
Logical Technology Component	Business Service	Provides platform for
Logical Technology Component	Logical Application Component	Serves
Logical Technology Component	Logical Technology Component	Decomposes
Logical Technology Component	Logical Technology Component	Is dependent on
Logical Technology Component	Physical Technology Component	Is realized by
Logical Technology Component	Technology Service	Supplies
Measure	Measure	Decomposes
Measure	Objective	Sets performance criteria for
Objective	Goal	Realizes
Objective	Measure	Is tracked against
Objective	Objective	Decomposes
Organization Unit	Actor	Contains
Organization Unit	Business Capability	Delivers
Organization Unit	Business Service	Owns and governs
Organization Unit	Course of Action	Participates in

Source Entity	Target Entity	Name
Organization Unit	Driver	Is motivated by
Organization Unit	Function	Enables
Organization Unit	Organization Unit	Decomposes
Organization Unit	Product	Delivers
Physical Application Component	Logical Application Component	Realizes
Physical Application Component	Physical Application Component	Decomposes
Physical Application Component	Physical Application Component	Communicates with
Physical Application Component	Physical Data Component	Used by
Physical Application Component	Physical Technology Component	Is served by
Physical Data Component	Logical Data Component	Realizes
Physical Data Component	Physical Application Component	Used by
Physical Data Component	Physical Data Component	Decomposes
Physical Technology Component	Logical Technology Component	Realizes
Physical Technology Component	Physical Application Component	Serves
Physical Technology Component	Physical Technology Component	Decomposes
Physical Technology Component	Physical Technology Component	Is dependent on
Process	Actor	Is produced by
Process	Actor	Supports
Process	Business Capability	Operationalizes
Process	Business Information	Is used by
Process	Business Service	Orchestrates
Process	Business Service	Decomposes
Process	Control	Is guided by
Process	Event	Generates
Process	Event	Resolves
Process	Function	Supports
Process	Function	Is realized by
Process	Process	Decomposes
Process	Process	Precedes, follows
Process	Product	Delivers
Process	Role	Involves
Process	Role	Is performed by
Process	Value Stream	Operationalizes
Product	Course of Action	Is produced by
Product	Organization Unit	Is produced by
Product	Process	Is produced by
Role	Actor	Is performed by
Role	Process	Participates in
Role	Process	Performs
Role	Role	Decomposes
Service Quality	Contract	Applies to
Service Quality	Service	Applies to
Technology Service	Application Service	Serves
Technology Service	Logical Technology Component	Is supplied by
Value Stream	Actor	Involves

Source Entity	Target Entity	Name
Value Stream	Actor	Is triggered by
Value Stream	Business Capability	Is enabled by
Value Stream	Course of Action	Is influenced by
Value Stream	Process	Is operationalized by
Work Package	Capability	Delivers

Chapter 3: Architectural Artifacts

This chapter discusses the concepts surrounding architecture artifacts and then describes the artifacts that are recommended to be created for each phase within the ADM.

3.1 Basic Concepts

Architectural artifacts are created in order to describe a system, solution, or state of the enterprise. The concepts discussed in this section have been adapted from more formal definitions contained in ISO/IEC/IEEE 42010: 2011 and ISO/IEC/IEEE 15288: 2015. They are illustrated in Figure 3-1.

The "environment" of a system is the context determining the setting and circumstances of all influences upon a system. The environment of a system includes developmental, technological, business, operational, organizational, political, economic, legal, regulatory, ecological, and social influences.

A "system" is a combination of interacting elements organized to achieve one or more stated purposes.

The "architecture" of a system is the fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution.

An "Architecture Description" is a work product used to express an architecture; a collection of architecture views and models that together document the architecture.

"Stakeholders" are individuals, teams, organizations, or classes thereof, having an interest in a system.

"Concerns" are interests in a system relevant to one or more of its stakeholders. Concerns may pertain to any aspect of the system's functioning, development, or operation, including considerations such as performance, reliability, security, distribution, and evolvability and may determine the acceptability of the system.

An "architecture view" is a representation of a system from the perspective of a related set of concerns. It consists of one or more architecture models of the system.

An "Architecture Model" is a representation of a subject of interest. A model provides a smaller scale, simplified, and/or abstract representation of the subject matter.

In capturing or representing the design of a system architecture, the architect will typically create one or more architecture models, possibly using different tools. An architecture view will comprise selected parts of one or more models, chosen so as to demonstrate to a particular stakeholder or group of stakeholders that their concerns are being adequately addressed in the design of the system architecture.

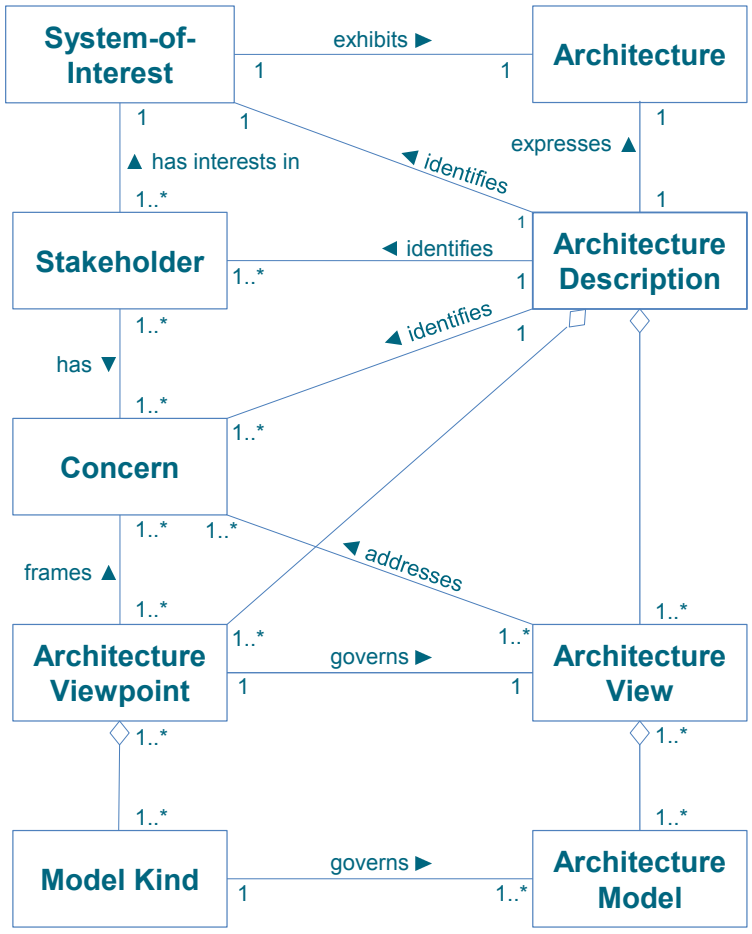

Figure 3-1 Basic Architectural Concepts

An "architecture viewpoint" is a specification of the conventions for a particular kind of architecture view. It can also be called the definition or schema for that kind of architecture view. It establishes the conventions for constructing, interpreting, and using an architecture view to address a specific concern (or set of concerns) about a system-of-interest.

A "Model Kind" establishes conventions for a type of modeling.

An architecture viewpoint references one or more model kinds; an architecture view incorporates one or more models.

A "viewpoint library" is a collection of the specifications of architecture viewpoints contained in the Reference Library portion of the Architecture Repository.

- An architecture view is what you see; an architecture viewpoint is where you are looking from — the vantage point or perspective that determines what you see

- Architecture viewpoints are generic, and can be stored in libraries for re-use; an architecture view is always specific to the architecture for which it is created
- Every architecture view has an associated architecture viewpoint that describes it, at least implicitly

  ISO/IEC/IEEE 42010: 2011 encourages architects to define architecture viewpoints explicitly. Making this distinction between the content and schema of a view may seem at first to be an unnecessary overhead, but it provides a mechanism for re-using architecture viewpoints across different architectures.

In summary, then, architecture views are representations of the overall architecture in terms meaningful to stakeholders. They enable the architecture to be communicated to and understood by the stakeholders, so they can verify that the system will address their concerns.

Concerns are often related to requirements. A concern can be a general requirement type, such as availability. It may lead to the definition of several particular requirements. It may be an interest related to some goal of a stakeholder. Identifying concerns helps ensure stakeholders' interests are addressed and requirements are identified. Associating concerns with general artifact types helps architects to select and develop particular artifacts for presentation to stakeholders.

3.1.1 Simple Example of an Architecture Viewpoint and Architecture View

For many architectures, a useful architecture viewpoint is that of business domains, which can be illustrated by an example from The Open Group itself.

The architecture viewpoint is specified as follows:

Architecture Viewpoint Element	Description
Stakeholders	Management Board, Chief Executive Officer
Concerns	Show the top-level relationships between US/UK geographical sites and business functions.
Modeling technique	Nested boxes diagram. Outer boxes = locations; inner boxes = business functions. Semantics of nesting = functions performed in the locations.

The corresponding architecture view of The Open Group (in 2017) is shown in Figure 3-2.

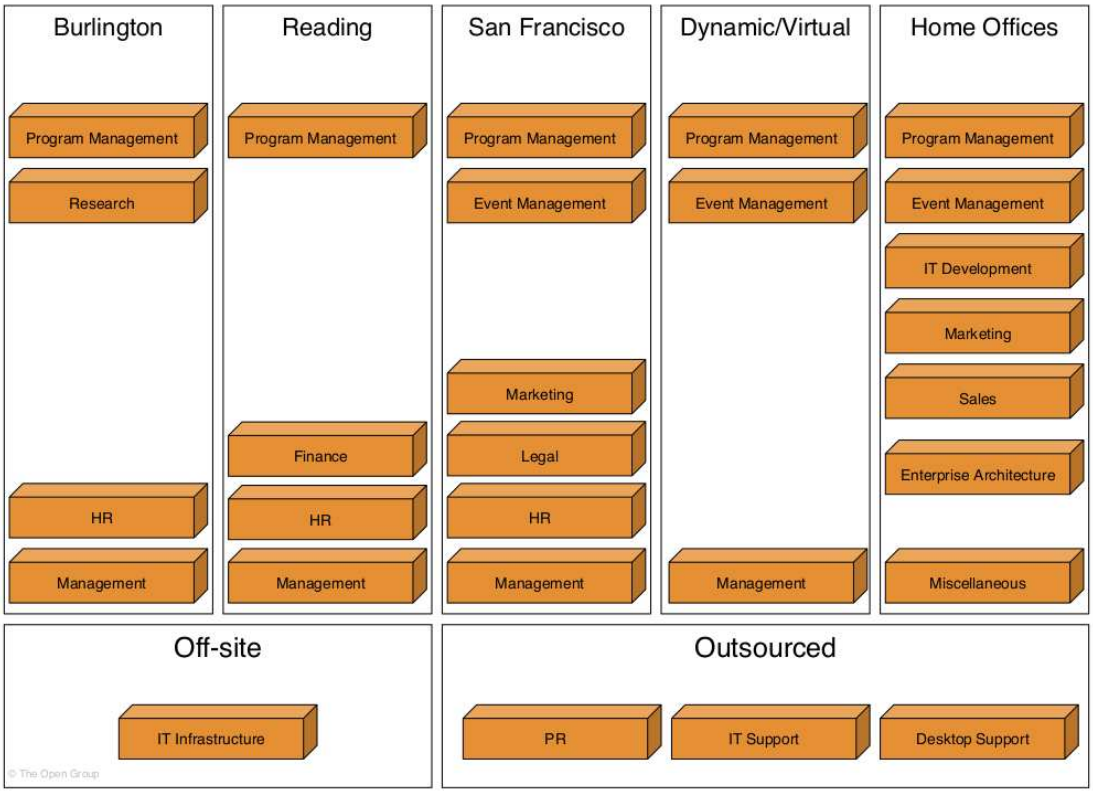

Figure 3-2 Example Architecture View — The Open Group Business Domains

3.2 Developing Architecture Views in the ADM

3.2.1 General Guidelines

The choice of which particular architecture views to develop is one of the key decisions that the architect has to make.

The architect has a responsibility for ensuring the completeness (fitness-for-purpose) of the architecture, in terms of adequately addressing all the pertinent concerns of its stakeholders; and the integrity of the architecture, in terms of connecting all the various views to each other, satisfactorily reconciling the conflicting concerns of different stakeholders, and showing the trade-offs made in so doing (as between security and performance, for example).

The choice has to be constrained by considerations of practicality, and by the principle of fitness-for-purpose (i.e., the architecture should be developed only to the point at which it is fit-for-purpose, and not reiterated *ad infinitum* as an academic exercise).

As explained in the TOGAF Standard — Architecture Development Method, the development of architecture views is an iterative process. The typical progression is from business to technology, using a technique such as business scenarios (see the TOGAF® Series Guide: Business

Scenarios) to properly identify all pertinent concerns; and from high-level overview to lower-level detail, continually referring back to the concerns and requirements of the stakeholders throughout the process.

Moreover, each of these progressions has to be made for two distinct environments: the existing environment (referred to as the baseline in the ADM) and the target environment. The architect must develop pertinent architecture views of both the Baseline Architecture and the Target Architecture. This provides the context for the gap analysis at the end of Phases B, C, and D of the ADM, which establishes the elements of the Baseline Architecture to be carried forward and the elements to be added, removed, or replaced.

This whole process is explained in the TOGAF Standard — ADM Techniques.

3.2.2 Architecture View Creation Process

As mentioned above, the TOGAF framework encourages but does not mandate the use of ISO/IEC/IEEE 42010: 2011. The following description therefore covers both the situation where ISO/IEC/IEEE 42010: 2011 has been adopted and where it has not.

ISO/IEC/IEEE 42010: 2011 itself does not require any specific process for developing architecture viewpoints or creating views from them. Where ISO/IEC/IEEE 42010: 2011 has been adopted and become well-established practice within an organization, it will often be possible to create the required views for a particular architecture by following these steps:

1. Refer to an existing library of architecture viewpoints
2. Select the appropriate architecture viewpoints (based on the stakeholders and concerns that need to be covered by views)
3. Generate views of the system by using the selected architecture viewpoints as templates

This approach can be expected to bring the following benefits:

- Less work for the architects (because the architecture viewpoints have already been defined and therefore the views can be created faster)
- Better comprehensibility for stakeholders (because the architecture viewpoints are already familiar)
- Greater confidence in the validity of the views (because their architecture viewpoints have a known track record)

However, situations can always arise in which an architecture view is needed for which no appropriate architecture viewpoint has been predefined. This is also the situation, of course, when an organization has not yet incorporated ISO/IEC/IEEE 42010: 2011 into its architecture practice and established a library of architecture viewpoints.

In each case, the architect may choose to develop a new architecture viewpoint that will cover the outstanding need, and then generate an architecture view from it. (This is ISO/IEC/IEEE 42010: 2011 recommended practice.) Alternatively, a more pragmatic approach can be equally successful: the architect can create an *ad hoc* architecture view for a specific system and later consider whether a generalized form of the implicit architecture viewpoint should be defined explicitly and saved in a library, so that it can be re-used. (This is one way of establishing a library of architecture viewpoints initially.)

Whatever the context, the architect should be aware that every architecture view has an

architecture viewpoint, at least implicitly, and that defining the architecture viewpoint in a systematic way (as recommended by ISO/IEC/IEEE 42010: 2011) will help in assessing its effectiveness; i.e., does the architecture viewpoint cover the relevant stakeholder concerns?

3.3 Views, Tools, and Languages

The need for architecture views, and the process of developing them following the ADM, is explained above. This section describes the relationships between architecture views, the tools used to develop and analyze them, and a standard language enabling interoperability between the tools.

3.3.1 Overview

In order to achieve the goals of completeness and integrity in an architecture, architecture views are usually developed, visualized, communicated, and managed using a tool.

In the current state of the market, different tools normally have to be used to develop and analyze different views of the architecture. It is highly desirable that an Architecture Description be encoded in a standard language, to enable a standard approach to the description of architecture semantics and their re-use among different tools.

An architecture viewpoint is also normally developed, visualized, communicated, and managed using a tool, and it is also highly desirable that standard architecture viewpoints (i.e., templates or schemas) be developed, so that different tools that deal in the same views can interoperate, the fundamental elements of an architecture can be re-used, and the Architecture Description can be shared among tools.

Issues relating to the evaluation of tools for architecture work are discussed in detail in the TOGAF Standard — Architecture Development Method.

3.4 Architecture Views and Architecture Viewpoints

3.4.1 Example of Architecture Views and Architecture Viewpoints

To illustrate the concepts of architecture views and architecture viewpoints, consider the example of a very simple airport system with two different stakeholders: the pilot and the air traffic controller.

One architecture view can be developed from the architecture viewpoint of the pilot, which addresses the pilot's concerns. Equally, another architecture view can be developed from the architecture viewpoint of the air traffic controller. Neither architecture view completely describes the system in its entirety, because the architecture viewpoint of each stakeholder constrains (and reduces) how each sees the overall system.

The architecture viewpoint of the pilot comprises some concerns that are not relevant to the controller, such as passengers and fuel, while the architecture viewpoint of the controller comprises some concerns not relevant to the pilot, such as other planes. There are also elements shared between the two architecture viewpoints, such as the communication model between the pilot and the controller, and the vital information about the plane itself.

An architecture viewpoint is a model (or description) of the information contained in a view. In

our example, one architecture viewpoint is the description of how the pilot sees the system, and the other architecture viewpoint is how the controller sees the system.

Pilots describe the system from their perspective, using a model of their position and vector toward or away from the runway. All pilots use this model, and the model has a specific language that is used to capture information and populate the model.

Controllers describe the system differently, using a model of the airspace and the locations and vectors of aircraft within the airspace. Again, all controllers use a common language derived from the common model in order to capture and communicate information pertinent to their architecture viewpoint.

Fortunately, when controllers talk with pilots, they use a common communication language. (In other words, the models representing their individual architecture viewpoints partially intersect.) Part of this common language is about location and vectors of aircraft, and is essential to safety.

So, in essence, each architecture viewpoint is an abstract model of how all the stakeholders of a particular type — all pilots, or all controllers — view the airport system.

Tools exist to assist stakeholders, especially when they are interacting with complex models such as the model of an airspace, or the model of air flight.

The interface to the human user of a tool is typically close to the model and language associated with the architecture viewpoint. The unique tools of the pilot are fuel, altitude, speed, and location indicators. The main tool of the controller is radar. The common tool is a radio.

To summarize from the above example, we can see that an architecture view can subset the system through the perspective of the stakeholder, such as the pilot *versus* the controller. This subset can be described by an abstract model called an architecture viewpoint, such as an air flight *versus* an air space model. This description of the architecture view is documented in a partially specialized language, such as "pilot-speak" *versus* "controller-speak". Tools are used to assist the stakeholders, and they interface with each other in terms of the language derived from the architecture viewpoint ("pilot-speak" *versus'* "controller-speak").

When stakeholders use common tools, such as the radio contact between pilot and controller, a common language is essential.

3.4.2 Architecture Views and Architecture Viewpoints in Enterprise Architecture

Now let us map this example to the Enterprise Architecture. Consider two stakeholders in a new small computing system: the users and the developers.

The users of the system have an architecture viewpoint that reflects their concerns when interacting with the system, and the developers of the system have a different architecture viewpoint. Architecture views that are developed to address either of the two architecture viewpoints are unlikely to exhaustively describe the whole system, because each perspective reduces how each sees the system.

The architecture viewpoint of the user is comprised of all the ways in which the user interacts with the system, not seeing any details such as applications or Database Management Systems (DBMS).

The architecture viewpoint of the developer is one of productivity and tools, and doesn't include things such as actual live data and connections with consumers.

However, there are things that are shared, such as descriptions of the processes that are

enabled by the system and/or communications protocols set up for users to communicate problems directly to development.

In this example, one architecture viewpoint is the description of how the user sees the system, and the other architecture viewpoint is how the developer sees the system. Users describe the system from their perspective, using a model of availability, response time, and access to information. All users of the system use this model, and the model has a specific language.

Developers describe the system differently than users, using a model of software connected to hardware distributed over a network, etc. However, there are many types of developers (database, security, etc.) of the system, and they do not have a common language derived from the model.

3.4.3 Need for a Common Language and Interoperable Tools for Architecture Description

Tools exist for both users and developers. Tools such as online help are there specifically for users, and attempt to use the language of the user. Many different tools exist for different types of developers, but they suffer from the lack of a common language that is required to bring the system together. It is difficult, if not impossible, in the current state of the tools market to have one tool interoperate with another tool.

Issues relating to the evaluation of tools for architecture work are discussed in detail in the TOGAF Standard — Architecture Development Method.

3.5 Conclusions

This section attempts to deal with views in a structured manner, but this is by no means a complete treatise on views.

In general, the TOGAF framework embraces the concepts and definitions presented in ISO/IEC/IEEE 42010: 2011, specifically the concepts that help guide the development of an architecture view and make the architecture view actionable. These concepts can be summarized as:

- Selecting a key stakeholder
- Understanding their concerns and generalizing/documenting those concerns
- Understanding how to model and deal with those concerns

3.6 Architectural Artifacts by ADM Phase

Catalog, Matrix, and Diagram Concept

The Enterprise Metamodel is used as a technique to structure architectural information in an ordered way so that it can be processed to meet the stakeholder needs. The majority of architecture stakeholders do not actually need to know what the architecture metamodel is and are only concerned with specific issues, such as "what functionality does this application support?", "which processes will be impacted by this project?", etc. In order to meet the needs of these stakeholders, the TOGAF concepts of building blocks, catalogs, matrices, and diagrams are used.

Building blocks are entities of a particular type within the metamodel (for example, a business service called "Purchase Order"). Building blocks carry metadata according to the metamodel, which supports query and analysis. For example, business services have a metadata attribute for owner, which allows a stakeholder to query all business services owned by a particular organization. Building blocks may also include dependent or contained entities as appropriate to the context of the architecture (for example, a business service called "Purchase Order" may implicitly include a number of processes, data entities, application components, etc.).

Catalogs are lists of building blocks of a specific type, or of related types, that are used for governance or reference purposes (for example, an organization chart, showing locations and actors). As with building blocks, catalogs carry metadata according to the metamodel, which supports query and analysis.

Matrices are grids that show relationships between two or more model entities. Matrices are used to represent relationships that are list-based rather than graphical in their usage (for example, a CRUD matrix showing which applications Create, Read, Update, and Delete a particular type of data is difficult to represent visually).

Diagrams are renderings of architectural content in a graphical format to allow stakeholders to retrieve the required information. Diagrams can also be used as a technique for graphically populating architecture content or for checking the completeness of information that has been collected. The TOGAF Content Framework defines a set of architecture diagrams to be created (e.g., organization chart). Each of these diagrams may be created several times for an architecture with different style or content coverage to suit stakeholder concerns.

Building blocks, catalogs, matrices, and diagrams are all concepts that are well supported by leading Enterprise Architecture tools. In environments where tools are used to model the architecture, such tools typically support mechanisms to search, filter, and query the Architecture Repository.

On-demand querying of the Architecture Repository (such as the business service ownership example mentioned above) can be used to generate *ad hoc* catalogs, matrices, and diagrams of the architecture. As this type of query is by nature required to be flexible, it is therefore not restricted or defined within the Enterprise Metamodel.

The interactions between metamodel, building blocks, diagrams, and stakeholders are shown in Figure 3-3. Figure 3-4 shows the artifacts that are associated with the TOGAF Enterprise Metamodel.

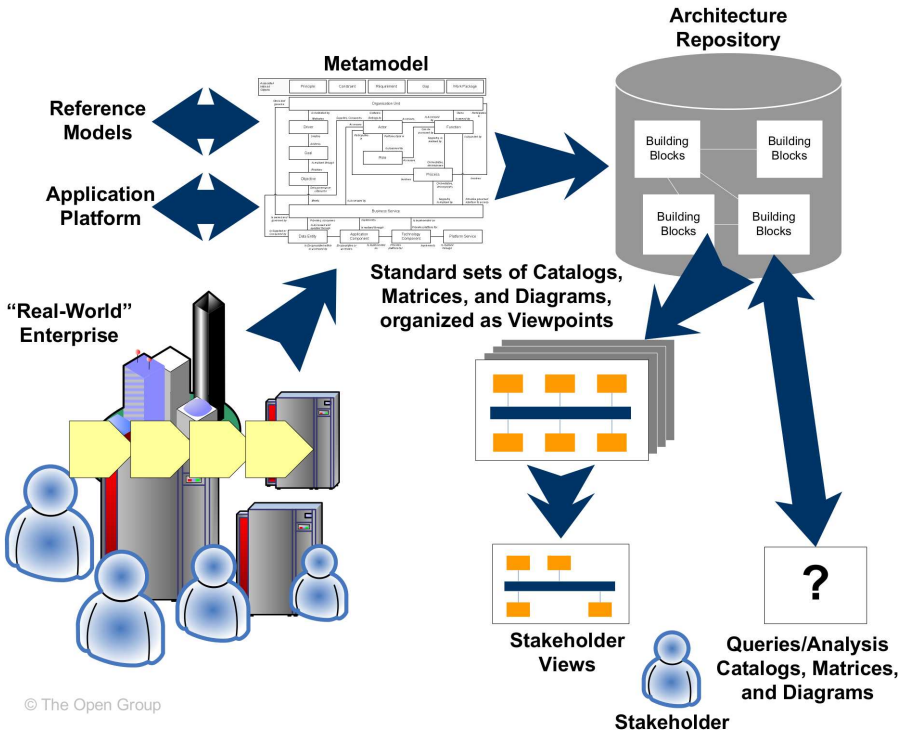

Figure 3-3 Interactions between Metamodel, Building Blocks, Diagrams, and Stakeholders

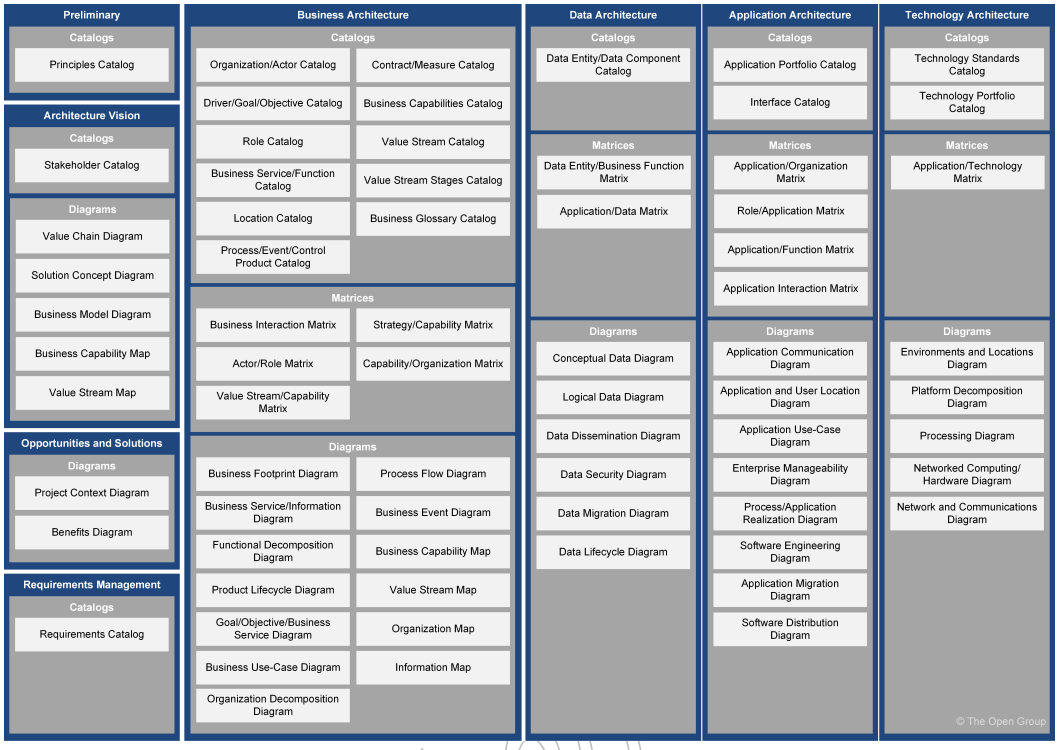

Figure 3-4 Artifacts Associated with the Enterprise Metamodel

The recommended artifacts for production in each ADM phase are as follows.

3.6.1 Preliminary Phase

The following describes catalogs, matrices, and diagrams that may be created within the Preliminary Phase, as described in the TOGAF Standard — Architecture Development Method.

Principles Catalog

The Principles catalog captures principles of the Business and Architecture Principles that describe what a "good" solution or architecture should look like. Principles are used to evaluate and agree an outcome for architecture decision points. Principles are also used as a tool to assist in architectural governance of change initiatives.

The Principles catalog contains the following metamodel entities:

- Principle

3.6.2 Phase A: Architecture Vision

The following describes catalogs, matrices, and diagrams that may be created within Phase A (Architecture Vision) as described in the TOGAF Standard — Architecture Development Method.

Stakeholder Catalog

The purpose of the Stakeholder catalog is to identify the stakeholders for the architecture engagement, their influence over the engagement, and their key questions, issues, or concerns that must be addressed by the architecture framework.

Understanding stakeholders and their requirements allows an architect to focus effort in areas that meet the needs of stakeholders (see the TOGAF Standard — ADM Techniques).

Due to the potentially sensitive nature of stakeholder mapping information and the fact that the Architecture Vision phase is intended to be conducted using informal modeling techniques, no specific metamodel entities will be used to generate a Stakeholder catalog.

Value Chain Diagram

A Value Chain diagram provides a high-level orientation view of an enterprise and how it interacts with the outside world. In contrast to the more formal organization map developed within Phase B (Business Architecture), the Value Chain diagram focuses on presentational impact.

The purpose of this diagram is to quickly on-board and align stakeholders for a particular change initiative, so that all participants understand the high-level functional and organizational context of the architecture engagement.

Solution Concept Diagram

A Solution Concept diagram provides a high-level orientation of the solution that is envisaged in order to meet the objectives of the architecture engagement. In contrast to the more formal and detailed architecture diagrams developed in the following phases, the solution concept represents a "pencil sketch" of the expected solution at the outset of the engagement.

This diagram may embody key objectives, requirements, and constraints for the engagement and also highlight work areas to be investigated in more detail with formal architecture modeling.

Its purpose is to quickly on-board and align stakeholders for a particular change initiative, so that all participants understand what the architecture engagement is seeking to achieve and how it is expected that a particular solution approach will meet the needs of the enterprise.

Business Model Diagram

A model describing the rationale for how an enterprise creates, delivers, and captures value.

Business Capability Map

A diagram that shows the business capabilities that an enterprise needs to meet its purposes.

Value Stream Map

A diagram representing an end-to-end collection of value-adding activities that create an overall result for a customer, stakeholder, or end user.

3.6.3 Phase B: Business Architecture

The following describes catalogs, matrices, and diagrams that may be created within Phase B (Business Architecture) as described in the TOGAF Standard — Architecture Development Method.

Organization/Actor Catalog

The purpose of the Organization/Actor catalog is to capture a definitive listing of all participants that interact with IT, including users and owners of IT systems.

The Organization/Actor catalog can be referenced when developing requirements in order to test for completeness.

For example, requirements for an application that services customers can be tested for completeness by verifying exactly which customer types need to be supported and whether there are any particular requirements or restrictions for user types.

The Organization/Actor catalog contains the following metamodel entities:

- Organization Unit
- Actor
- Location (may be included in this catalog if an independent Location catalog is not maintained)

Driver/Goal/Objective Catalog

The purpose of the Driver/Goal/Objective catalog is to provide a cross-organizational reference of how an organization meets its drivers in practical terms through goals, objectives, and (optionally) measures.

Publishing a definitive breakdown of drivers, goals, and objectives allows change initiatives within the enterprise to identify synergies across the organization (e.g., multiple organizations attempting to achieve similar objectives), which in turn allow stakeholders to be identified and related change initiatives to be aligned or consolidated.

The Driver/Goal/Objective catalog contains the following metamodel entities:

- Organization Unit
- Driver
- Goal
- Objective

- Measure (may optionally be included)

Role Catalog

The purpose of the Role catalog is to provide a listing of all authorization levels or zones within an enterprise. Frequently, application security or behavior is defined against locally understood concepts of authorization that create complex and unexpected consequences when combined on the user desktop.

If roles are defined, understood, and aligned across organizations and applications, this allows for a more seamless user experience and generally more secure applications, as administrators do not need to resort to workarounds in order to enable users to carry out their jobs.

In addition to supporting security definition for the enterprise, the Role catalog also forms a key input to identifying organizational change management impacts, defining job functions, and executing end-user training.

As each role implies access to a number of business functions, if any of these business functions are impacted then change management will be required, organizational responsibilities may need to be redefined, and retraining may be needed.

The Role catalog contains the following metamodel entities:

- Role

Business Service/Function Catalog

The purpose of the Business Service/Function catalog is to provide a functional decomposition in a form that can be filtered, reported on, and queried, as a supplement to graphical Functional Decomposition diagrams.

The Business Service/Function catalog can be used to identify capabilities of an organization and to understand the level that governance is applied to the functions of an organization. This functional decomposition can be used to identify new capabilities required to support business change or may be used to determine the scope of change initiatives, applications, or technology components.

The Business Service/Function catalog contains the following metamodel entities:

- Organization Unit
- Business Function
- Business Service
- Application Service (may optionally be included here)

Location Catalog

The Location catalog provides a listing of all locations where an enterprise carries out business operations or houses architecturally relevant assets, such as data centers or end-user computing equipment.

Maintaining a definitive list of locations allows change initiatives to quickly define a location scope and to test for completeness when assessing current landscapes or proposed target solutions. For example, a project to upgrade desktop operating systems will need to identify all locations where desktop operating systems are deployed.

Similarly, when new systems are being implemented a diagram of locations is essential in order

to develop appropriate deployment strategies that comprehend both user and application location and identify location-related issues, such as internationalization, localization, timezone impacts on availability, distance impacts on latency, network impacts on bandwidth, and access.

The Location catalog contains the following metamodel entities:

- Location

Process/Event/Control/Product Catalog

The Process/Event/Control/Product catalog provides a hierarchy of processes, events that trigger processes, outputs from processes, and controls applied to the execution of processes. This catalog provides a supplement to any Process Flow diagrams that are created and allows an enterprise to filter, report, and query across organizations and processes to identify scope, commonality, or impact.

For example, the Process/Event/Control/Product catalog allows an enterprise to see relationships of processes to sub-processes in order to identify the full chain of impacts resulting from changing a high-level process.

The Process/Event/Control/Product catalog contains the following metamodel entities:

- Process
- Event
- Control
- Product

Contract/Measure Catalog

The Contract/Measure catalog provides a listing of all agreed service contracts and the measures attached to those contracts. It forms the master list of service levels agreed to across the enterprise.

The Contract/Measure catalog contains the following metamodel entities:

- Business Service
- Application Service (optionally)
- Contract
- Measure

Business Capabilities Catalog

A listing of abilities that a business may possess to achieve specific purposes.

Value Stream Catalog

A listing of end-to-end collections of value-adding activities that create an overall result for a customer, stakeholder, or end user.

Value Stream Stages Catalog

A listing of end-to-end collections of the different stages for the value-adding activities that create an overall result for customer, stakeholder, or end user.

The Value Stream Stages catalog includes the following metamodel entities:

- Business Capability
- Value Stream

Business Glossary Catalog

This catalog contains the name and unambiguous description of all elements contained in or interacting with the Enterprise Architecture.

Optional information could include element enterprise synonyms, abbreviations/acronyms, and relationships with other elements. This catalog provides the semantics to be used by all analysts (e.g., business, information, and system) for their architecture products, including information/data models, and evolves throughout the ADM.

Business Interaction Matrix

The purpose of this matrix is to depict the relationship interactions between organizations and business functions across the enterprise.

Understanding business interaction of an enterprise is important as it helps to highlight value chain and dependencies across organizations.

The Business Interaction matrix shows the following metamodel entities and relationships:

- Organization
- Business Function
- Business Service
- Business Service *communicates with* Business Service relationships
- Business Service *is dependent on* Business Service relationships

Actor/Role Matrix

The purpose of this matrix is to show which actors perform which roles, supporting definition of security and skills requirements.

Understanding Actor-to-Role relationships is a key supporting tool in definition of training needs, user security settings, and organizational change management.

The Actor/Role matrix shows the following metamodel entities and relationships:

- Actor
- Role
- Actor *performs* Role relationships

Value Stream/Capability Matrix

The purpose of this matrix is to show the capabilities required to support each stage of a value stream.

Strategy/Capability Matrix

The purpose of this matrix is to show the capabilities required to support specific strategy statements.

Capability/Organization Matrix

The purpose of this matrix is to show the organization elements that implement each capability. The Capability/Organization matrix includes the following metamodel entities:

- Business Capability
- Value Stream
- Organization Unit

Business Footprint Diagram

A Business Footprint diagram describes the links between business goals, organizational units, business functions, and services, and maps these functions to the technical components delivering the required capability.

A Business Footprint diagram provides a clear traceability between a technical component and the business goal that it satisfies, while also demonstrating ownership of the services identified.

A Business Footprint diagram demonstrates only the key facts linking organization unit functions to delivery services and is utilized as a communication platform for senior-level (CxO) stakeholders.

Business Service/Information Diagram

The Business Service/Information diagram shows the information needed to support one or more business services. The Business Service/Information diagram shows what data is consumed by or produced by a business service and may also show the source of information.

The Business Service/Information diagram shows an initial representation of the information present within the architecture and therefore forms a basis for elaboration and refinement within Phase C (Data Architecture).

Functional Decomposition Diagram

The purpose of the Functional Decomposition diagram is to show on a single page the capabilities of an organization that are relevant to the consideration of an architecture. By examining the capabilities of an organization from a functional perspective, it is possible to quickly develop models of what the organization does without being dragged into extended debate on how the organization does it.

Once a basic Functional Decomposition diagram has been developed, it becomes possible to layer heat maps on top of this diagram to show scope and decisions. For example, the capabilities to be implemented in different phases of a change program.

Product Lifecycle Diagram

This diagram shows the possible state transitions of a business product, from its creation or receipt to its sale, disposal, or destruction.

The product may be a product of any kind (physical, software, or service).

Goal/Objective/Business Service Diagram

The purpose of a Goal/Objective/Business Service diagram is to define the ways in which a business service contributes to the achievement of a business vision or strategy.

Business services are associated with the drivers, goals, objectives, and measures that they support, allowing the enterprise to understand which business services contribute to similar aspects of business performance. The Goal/Objective/Business Service diagram also provides qualitative input on what constitutes high performance for a particular business service.

Business Use-Case Diagram

A Business Use-Case diagram displays the relationships between consumers and providers of business services. Business services are consumed by actors or other business services and the Business Use-Case diagram provides added richness in describing business capability by illustrating how and when that capability is used.

The purpose of the Business Use-Case diagram is to help to describe and validate the interaction between actors and their roles to processes and functions. As the architecture progresses, the use-case can evolve from the business level to include data, application, and technology details. Architectural business use-cases can also be re-used in systems design work.

Organization Decomposition Diagram

An Organization Decomposition diagram describes the links between actor, roles, and location within an organization tree.

An organization map should provide a chain of command of owners and decision-makers in the organization. Although it is not the intent of the Organization Decomposition diagram to link goal to organization, it should be possible to intuitively link the goals to the stakeholders from the Organization Decomposition diagram.

Process Flow Diagram

The purpose of the Process Flow diagram is to depict all models and mappings related to the process metamodel entity.

Process Flow diagrams show sequential flow of control between activities and may utilize swim-lane techniques to represent ownership and realization of process steps. For example, the application that supports a process step may be shown as a swim-lane.

In addition to showing a sequence of activity, process flows can also be used to detail the controls that apply to a process, the events that trigger or result from completion of a process, and also the products that are generated from process execution.

Process Flow diagrams are useful in elaborating the architecture with subject specialists, as they allow the specialist to describe "how the job is done" for a particular function. Through this process, each process step can become a more fine-grained function and can then in turn be elaborated as a process.

Business Event Diagram

The purpose of the Business Event diagram is to depict the relationship between events and process.

Certain events — such as arrival of certain information (e.g., customer submits sales order) or a certain point in time (e.g., end of fiscal quarter) — cause work and certain actions need to be undertaken within the business. These are often referred to as "business events" or simply "events" and are considered as triggers for a process. It is important to note that the event has to trigger a process and generate a business response or result.

Business Capability Map

A diagram that shows the business capabilities that an enterprise needs to meet its purposes.

Business capabilities may be decomposed into sub-capabilities.

Value Stream Map

A diagram representing an end-to-end collection of value-adding activities that create an overall result for a customer, stakeholder, or end user.

The value stream map includes the following metamodel entities:

- Business Capability
- Value Stream

Organization Map

A diagram showing the relationships between the primary entities that make up the enterprise, its partners, and stakeholders.

Information Map

A collection of information concepts and their relationships to one another. Information concepts, in effect, reflect the business vocabulary; e.g., client, account, or product. Mapping information in the Business Architecture starts with listing those elements that matter most to the business as well as how they are described in business terms.

3.6.4 Phase C: Data Architecture

The following describes catalogs, matrices, and diagrams that may be created within Phase C (Data Architecture) as described in the TOGAF Standard — Architecture Development Method.

Data Entity/Data Component Catalog

This catalog identifies and maintains a list of the data used across the enterprise, relating data entities to data components showing how data entities are structured.

Its purpose is to:

- Identify all data used in the enterprise
- Encourage effective data use

The Data Entity/Data Component catalog contains the following metamodel entities:

- Data Entity
- Logical Data Component
- Physical Data Component

Data Entity/Business Function Matrix

The purpose of the Data Entity/Business Function matrix is to depict the relationship between data entities and business functions within the enterprise. Business functions are supported by business services with explicitly defined boundaries and will be supported and realized by business processes. The mapping of the Data Entity-Business Function relationship enables the following to take place:

- Assign ownership of data entities to organizations
- Understand the data and information exchange requirements business services
- Support the gap analysis and determine whether any data entities are missing and need to be created
- Define application of origin, application of record, and application of reference for data entities
- Enable development of data governance programs across the enterprise (establish data steward, develop data standards pertinent to the business function, etc.)

The Data Entity/Business Function matrix shows the following entities and relationships:

- Data Entity
- Business Function
- Data Entity relationship to owning Organization Unit

Application/Data Matrix

The purpose of the Application/Data matrix is to depict the relationship between applications (i.e., application components) and the data entities that are accessed and updated by them.

Applications will create, read, update, and delete specific data entities that are associated with them. For example, a Customer Relationship Management (CRM) application will create, read, update, and delete customer entity information.

The data entities in a package/packaged services environment can be classified as master data, reference data, transactional data, content data, and historical data. Applications that operate on the data entities include transactional applications, information management applications, and business warehouse applications.

The mapping of the Application Component-Data Entity relationship is an important step as it enables the following to take place:

- Assign access of data to specific applications in the organization
- Understand the degree of data duplication within different applications, and the scale of the data lifecycle
- Understand where the same data is updated by different applications

- Support the gap analysis and determine whether any of the applications are missing and as a result need to be created

The Application/Data matrix is a two-dimensional table with Logical Application Component on one axis and Data Entity on the other axis.

Conceptual Data Diagram

The key purpose of the Conceptual Data diagram is to depict the relationships between critical data entities within the enterprise. This diagram is developed to address the concerns of business stakeholders.

Techniques used include:

- Entity relationship models
- Simplified UML class diagrams

Logical Data Diagram

The key purpose of the Logical Data diagram is to show logical views of the relationships between critical data entities within the enterprise. This diagram is developed to address the concerns of:

- Application developers
- Database designers

Data Dissemination Diagram

The purpose of the Data Dissemination diagram is to show the relationship between data entity, business service, and application components. The diagram shows how the logical entities are to be physically realized by application components. This allows effective sizing to be carried out and the IT footprint to be refined. Moreover, by assigning business value to data, an indication of the business criticality of application components can be gained.

Additionally, the diagram may show data replication and application ownership of the master reference for data. In this instance, it can show two copies and the master-copy relationship between them. This diagram can include services; that is, services encapsulate data and they reside in an application, or services that reside on an application and access data encapsulated within the application.

Data Security Diagram

This diagram shows which data is accessed by which roles, organization units, and applications. It may be shown as a diagram or presented as a matrix.

Its purpose is to:

- Demonstrate compliance with data privacy laws and regulations
- Show security threats arising as a result of data access
- Show which parties outside of the organization have access to data
- Show security measures applied to data access

Data Migration Diagram

This diagram shows how data is extracted from baseline or source location(s) and loaded into target location(s). It may show where data is transformed and/or cleansed on the way. In Phase C of the ADM, the diagram is likely to be at an overview level. Later, it may be elaborated to show which source data items map to which target data items.

Data Lifecycle Diagram

The Data Lifecycle diagram is an essential part of managing business data throughout its lifecycle from conception until disposal within the constraints of the business process.

The data is considered as an entity in its own right, decoupled from business process and activity. Each change in state is represented on the diagram which may include the event or rules that trigger that change in state.

The separation of data from process allows common data requirements to be identified which enables resource sharing to be achieved more effectively.

3.6.5 Phase C: Application Architecture

The following describes catalogs, matrices, and diagrams that may be created within Phase C (Application Architecture) as described in the TOGAF Standard — Architecture Development Method.

Application Portfolio Catalog

The purpose of this catalog is to identify and maintain a list of all the applications in the enterprise. This list helps to define the horizontal scope of change initiatives that may impact particular kinds of applications. An agreed Application Portfolio allows a standard set of applications to be defined and governed.

The Application Portfolio catalog provides a foundation on which to base the remaining matrices and diagrams. It is typically the start point of the Application Architecture phase.

The Application Portfolio catalog contains the following metamodel entities:

- Application Service
- Logical Application Component
- Physical Application Component

Interface Catalog

The purpose of the Interface catalog is to scope and document the interfaces between applications to enable the overall dependencies between applications to be scoped as early as possible.

Applications will create, read, update, and delete data within other applications; this will be achieved by some kind of interface, whether via a batch file that is loaded periodically, a direct connection to another application's database, or via some form of API or web service.

The mapping of the Application Component-Application Component entity relationship is an important step as it enables the following to take place:

- Understand the degree of interaction between applications, identifying those that are central in terms of their dependencies on other applications
- Understand the number and types of interfaces between applications
- Understand the degree of duplication of interfaces between applications
- Identify the potential for simplification of interfaces when considering the target Application Portfolio
- Support the gap analysis and determine whether any of the applications are missing and as a result need to be created

The Interface catalog contains the following metamodel entities:

- Logical Application Component
- Physical Application Component
- Application *communicates with* application relationship

Application/Organization Matrix

The purpose of this matrix is to depict the relationship between applications and organizational units within the enterprise.

Business operations are performed by organizational units. Some of the operations and services performed by those organizational units will be supported by applications. The mapping of the Application Component-Organization Unit relationship is an important step as it enables the following to take place:

- Assign usage of applications to the organization units that perform business operations
- Understand the application support requirements of the business services and processes carried out by an organization unit
- Support the gap analysis and determine whether any of the applications are missing and as a result need to be created
- Define the application set used by a particular organization unit

The Application/Organization matrix is a two-dimensional table with Logical/Physical Application Component on one axis and Organization Unit on the other axis.

The relationship between these two entities is a composite of a number of metamodel relationships that need validating:

- Organization Units *own* Services
- Actors that *belong to* Organization Units *use* Services
- Services are *realized by* Logical/Physical Application Components

Role/Application Matrix

The purpose of the Role/Application matrix is to depict the relationship between applications and the business roles that use them within the enterprise.

People in an organization interact with applications. During this interaction, these people assume a specific role to perform a task; for example, product buyer.

The mapping of the Application Component-Role relationship is an important step as it enables the following to take place:

- Assign usage of applications to the specific roles in the organization
- Understand the application security requirements of the business services and processes supporting the function, and check these are in line with current policy
- Support the gap analysis and determine whether any of the applications are missing and as a result need to be created
- Define the application set used by a particular business role; essential in any move to role-based computing

The Role/Application matrix is a two-dimensional table with Logical Application Component on one axis and Role on the other axis.

Application/Function Matrix

The purpose of the Application/Function matrix is to depict the relationship between applications and business functions within the enterprise.

Business functions are performed by organizational units. Some of the business functions and services will be supported by applications. The mapping of the Application Component-Function relationship is an important step as it enables the following to take place:

- Assign usage of applications to the business functions that are supported by them
- Understand the application support requirements of the business services and processes carried out
- Support the gap analysis and determine whether any of the applications are missing and as a result need to be created
- Define the application set used by a particular business function

The Application/Function matrix is a two-dimensional table with Logical Application Component on one axis and Function on the other axis.

The relationship between these two entities is a composite of a number of metamodel relationships that need validating:

- Function *is bounded by* Service
- Services are *realized by* Logical/Physical Application Components

Application Interaction Matrix

The purpose of the Application Interaction matrix is to depict communications relationships between applications.

The mapping of the application interactions shows in matrix form the equivalent of the Interface Catalog or an Application Communication diagram.

The Application Interaction matrix is a two-dimensional table with Application Service, Logical Application Component, and Physical Application Component on both the rows and the columns of the table.

The relationships depicted by this matrix include:

- Application Service *consumes* Application Service
- Logical Application Component *communicates with* Logical Application Component
- Physical Application Component *communicates with* Physical Application Component

Application Communication Diagram

The purpose of the Application Communication diagram is to depict all models and mappings related to communication between applications in the metamodel entity.

It shows application components and interfaces between components. Interfaces may be associated with data entities where appropriate. Applications may be associated with business services where appropriate. Communication should be logical and should only show intermediary technology where it is architecturally relevant.

Application and User Location Diagram

The Application and User Location diagram shows the geographical distribution of applications. It can be used to show where applications are used by the end user; the distribution of where the host application is executed and/or delivered in thin client scenarios; the distribution of where applications are developed, tested, and released; etc.

Analysis can reveal opportunities for rationalization, as well as duplication and/or gaps.

The purpose of this diagram is to clearly depict the business locations from which business users typically interact with the applications, but also the hosting location of the application infrastructure.

The diagram enables:

- Identification of the number of package instances needed to sufficiently support the user population that may be spread out geographically
- Estimation of the number and type of user licenses for the package or other software
- Estimation of the level of support needed for the users and the location of the support center
- Selection of system management tools, structure, and management system required to support the enterprise users/customers/partners both locally and remotely
- Appropriate planning for the technological components of the business, namely server sizing and network bandwidth, etc.

- Performance considerations while implementing application and technology architecture solutions

Users typically interact with applications in a variety of ways; for example:

- To support the operations of the business day-to-day
- To participate in the execution of a business process
- To access information (look-up, read)
- To develop the application
- To administer and maintain the application

Application Use-Case Diagram

An Application Use-Case diagram displays the relationships between consumers and providers of application services. Application services are consumed by actors or other application services and the Application Use-Case diagram provides added richness in describing application functionality by illustrating how and when that functionality is used.

The purpose of the Application Use-Case diagram is to help to describe and validate the interaction between actors and their roles with applications. As the architecture progresses, the use-case can evolve from functional information to include technical realization detail.

Application use-cases can also be re-used in more detailed systems design work.

Enterprise Manageability Diagram

The Enterprise Manageability diagram shows how one or more applications interact with application and technology components that support the operational management of a solution.

This diagram is really a filter on the Application Communication diagram, specifically for enterprise management class software.

Analysis can reveal duplication and gaps, and opportunities in the IT service management operation of an organization.

Process/Application Realization Diagram

The purpose of the Process/Application Realization diagram is to clearly depict the sequence of events when multiple applications are involved in executing a business process.

It enhances the Application Communication diagram by augmenting it with any sequencing constraints, and hand-off points between batch and real-time processing.

It would identify complex sequences that could be simplified, and identify possible rationalization points in the architecture in order to provide more timely information to business users. It may also identify process efficiency improvements that may reduce interaction traffic between applications.

Software Engineering Diagram

The Software Engineering diagram breaks applications into packages, modules, services, and operations from a development perspective.

It enables more detailed impact analysis when planning migration stages, and analyzing opportunities and solutions.

It is ideal for application development teams and application management teams when managing complex development environments.

Application Migration Diagram

The Application Migration diagram identifies application migration from baseline to target application components. It enables a more accurate estimation of migration costs by showing precisely which applications and interfaces need to be mapped between migration stages.

It would identify temporary applications, staging areas, and the infrastructure required to support migrations (for example, parallel run environments, etc).

Software Distribution Diagram

The Software Distribution diagram shows how application software is structured and distributed across the estate. It is useful in systems upgrade or application consolidation projects.

This diagram shows how physical applications are distributed across physical technology and the location of that technology.

This enables a clear view of how the software is hosted, but also enables managed operations staff to understand how that application software is maintained once installed.

3.6.6 Phase D: Technology Architecture

The following section describes catalogs, matrices, and diagrams that may be created within Phase D (Technology Architecture) as described in the TOGAF Standard — Architecture Development Method.

Technology Standards Catalog

The Technology Standards catalog documents the agreed standards for technology across the enterprise covering technologies, and versions, the technology lifecycles, and the refresh cycles for the technology.

Depending upon the organization, this may also include location or business domain-specific standards information.

This catalog provides a snapshot of the enterprise standard technologies that are or can be deployed, and also helps identify the discrepancies across the enterprise.

The Technology Standards catalog contains the following metamodel entities:

- Technology Service
- Logical Technology Component

- Physical Technology Component

If technology standards are currently in place, apply these to the Technology Portfolio catalog to gain a baseline view of compliance with technology standards.

Technology Portfolio Catalog

The purpose of this catalog is to identify and maintain a list of all the technology in use across the enterprise, including hardware, infrastructure software, and application software. An agreed technology portfolio supports the lifecycle management of technology products and versions and also forms the basis for the definition of technology standards.

The Technology Portfolio catalog provides a foundation on which to base the remaining matrices and diagrams. It is typically the start point of the Technology Architecture phase.

Technology registries and repositories also provide input into this catalog from a baseline and target perspective.

Technologies in the catalog should be classified against the defined taxonomy in use in the enterprise, such as the TOGAF TRM — see the TOGAF® Series Guide: The TOGAF® Technical Reference Model (TRM) — adapted as necessary to fit the classification of technology products in use.

The Technology Portfolio catalog contains the following metamodel entities:

- Technology Service
- Logical Technology Component
- Physical Technology Component

Application/Technology Matrix

The Application/Technology matrix documents the mapping of applications to technology platform.

This matrix should be aligned with and complement one or more platform decomposition diagrams.

The Application/Technology matrix shows:

- Logical/Physical Application Components
- Services, Logical Technology Components, and Physical Technology Components
- Physical Technology Component *realizes* Physical Application Component relationships

Environments and Locations Diagram

The Environments and Locations diagram depicts which locations host which applications, identifies what technologies and/or applications are used at which locations, and finally identifies the locations from which business users typically interact with the applications.

This diagram should also show the existence and location of different deployment environments, including non-production environments, such as development and pre-production.

Platform Decomposition Diagram

The Platform Decomposition diagram depicts the technology platform that supports the operations of the Information Systems Architecture. The diagram covers all aspects of the infrastructure platform and provides an overview of the enterprise's technology platform. The diagram can be expanded to map the technology platform to appropriate application components within a specific functional or process area. This diagram may show specification details, such as product versions, number of CPUs, etc. or simply could be an informal "eye-chart" providing an overview of the technical environment.

The diagram should clearly show the enterprise applications. The technology platform for each application area can further be decomposed as follows:

- Hardware:
  - Logical Technology Components (with attributes)
  - Physical Technology Components (with attributes)
- Software:
  - Logical Technology Components (with attributes)
  - Physical Technology Components (with attributes)

Depending upon the scope of the Enterprise Architecture work, additional technology cross-platform information (e.g., communications, telco, and video information) may be addressed.

Processing Diagram

The Processing diagram focuses on deployable units of code/configuration and how these are deployed onto the technology platform. A deployment unit represents the grouping of business capability, service, or application components. The Processing diagram addresses the following:

- Which set of application components need to be grouped to form a deployment unit
- How one deployment unit connects/interacts with another (LAN, WAN, and the applicable protocols)
- How application configuration and usage patterns generate load or capacity requirements for different technology components

The organization and grouping of deployment units depends on separation concerns of the presentation, business logic, and data store layers and service-level requirements of the components. For example, the presentation layer deployment unit is grouped based on the following:

- Application components that provide user interface or user access functions
- Application components that are differentiated by location and user roles

There are several considerations to determine how application components are grouped together. Each deployment unit is made up of sub-units, such as:

- **Installation**: part that holds the executable code or package configuration (in case of packages)
- **Execution**: application component with its associated state at run time

- **Persistence**: data that represents the persistent state of the application component

Finally, these deployment units are deployed on either dedicated or shared technology components (workstation, web server, application server, or database server, etc.). It is important to note that technology processing can influence and have implications on the services definition and granularity.

Networked Computing/Hardware Diagram

Starting with the transformation to client-server systems from mainframes and later with the advent of e-Business and J2EE, large enterprises moved predominantly into a highly network-based distributed network computing environment with firewalls and demilitarized zones. Currently, most of the applications have a web front-end and, looking at the deployment architecture of these applications, it is very common to find three distinct layers in the network landscape; namely a web presentation layer, a business logic or application layer, and a back-end data store layer. It is common practice for applications to be deployed and hosted in a shared and common infrastructure environment.

So it becomes highly critical to document the mapping between logical applications and the technology components (e.g., server) that supports the application both in the development and production environments. The purpose of this diagram is to show the "as deployed" logical view of logical application components in a distributed network computing environment. The diagram is useful for the following reasons:

- Enable understanding of which application is deployed where in the distributed network computing environment
- Establishing authorization, security, and access to these technology components
- Understand the Technology Architecture that supports the applications during problem resolution and troubleshooting
- Isolate performance problems encountered by applications, determine whether it is application code-related or technology platform-related, and perform necessary upgrade to specific physical technology components
- Identify areas of optimization as and when newer technologies are available which will eventually reduce cost
- Enable application/technology auditing and prove compliance with enterprise technology standards
- Serve as an important tool to introduce changes to the Technology Architecture, thereby supporting effective change management
- Establish traceability and changing application end-point address while moving application either from a shared environment to a dedicated environment or *vice versa*

The scope of the diagram can be appropriately defined to cover a specific application, business function, or the entire enterprise. If chosen to be developed at the enterprise level, then the network computing landscape can be depicted in an application-agnostic way as well.

Network and Communications Diagram

The Network and Communications diagram describes the means of communication — the method of sending and receiving information — between these assets in the Technology Architecture; insofar as the selection of package solutions in the preceding architectures put specific requirements on the communications between the applications.

The Network and Communications diagram will take logical connections between client and server components and identify network boundaries and network infrastructure required to physically implement those connections. It does not describe the information format or content, but will address protocol and capacity issues.

3.6.7 Phase E: Opportunities and Solutions

The following section describes catalogs, matrices, and diagrams that may be created within Phase E (Opportunities & Solutions) as described in the TOGAF Standard — Architecture Development Method.

Project Context Diagram

A Project Context diagram shows the scope of a work package to be implemented as a part of a broader transformation roadmap. The Project Context diagram links a work package to the organizations, functions, services, processes, applications, data, and technology that will be added, removed, or impacted by the project.

The Project Context diagram is also a valuable tool for project portfolio management and project mobilization.

Benefits Diagram

The Benefits diagram shows opportunities identified in an architecture definition, classified according to their relative size, benefit, and complexity. This diagram can be used by stakeholders to make selection, prioritization, and sequencing decisions on identified opportunities.

3.6.8 Requirements Management

The following section describes catalogs, matrices, and diagrams that may be created within the Requirements Management phase as described in the TOGAF Standard — Architecture Development Method.

Requirements Catalog

The Requirements catalog captures things that the enterprise needs to do to meet its objectives. Requirements generated from architecture engagements are typically implemented through change initiatives identified and scoped during Phase E (Opportunities & Solutions). Requirements can also be used as a quality assurance tool to ensure that a particular architecture is fit-for-purpose (i.e., the architecture can meet all identified requirements).

The Requirements catalog contains the following metamodel entities:

- Requirement
- Assumption
- Constraint
- Gap

Chapter 4: Architecture Deliverables

This chapter provides descriptions of deliverables referenced in the ADM.

4.1 Introduction

This chapter defines the deliverables that will typically be consumed and produced across the TOGAF ADM cycle. As deliverables are typically the contractual or formal work products of an architecture project, it is likely that these deliverables will be constrained or altered by any overarching project or process management for the enterprise (such as CMMI®, PRINCE2®, PMBOK®, or MSP®).

This chapter therefore is intended to provide a typical baseline of architecture deliverables in order to better define the activities required in the ADM and act as a starting point for tailoring within a specific organization.

The TOGAF Content Framework (see Chapter 1) identifies deliverables that are produced as outputs from executing the ADM cycle and potentially consumed as inputs at other points in the ADM. Other deliverables may be produced elsewhere and consumed by the ADM.

Deliverables produced by executing the ADM are shown in the table below.

Deliverable	Output from...	Input to...
Architecture Building Blocks (see Section 4.2.1)	F, H	A, B, C, D, E
Architecture Contract (see Section 4.2.2)	G	G, H
Architecture Definition Document (see Section 4.2.3)	A, B, C, D, E, F	B, C, D, E, F, G, H
Architecture Principles (see Section 4.2.4)	Preliminary, A, B, C, D	Preliminary, A, B, C, D, E, F, G, H
Architecture Repository (see Section 4.2.5)	Preliminary	Preliminary, A, B, C, D, E, F, G, H, Requirements Management
Architecture Requirements Specification (see Section 4.2.6)	B, C, D, E, F, Requirements Management	C, D, Requirements Management
Architecture Roadmap (see Section 4.2.7)	B, C, D, E, F	B, C, D, E, F
Architecture Vision (see Section 4.2.8)	A, E	B, C, D, E, F, G, H, Requirements Management
Business Principles, Business Goals, and Business Drivers	Preliminary, A, B	A, B

Deliverable	Output from...	Input to...
(see Section 4.2.9)		
Capability Assessment (see Section 4.2.10)	A, E	B, C, D, E, F
Change Request (see Section 4.2.11)	F, G, H	—
Communications Plan (see Section 4.2.12)	A	B, C, D, E, F
Compliance Assessment (see Section 4.2.13)	G	H
Implementation and Migration Plan (see Section 4.2.14)	E, F	F
Implementation Governance Model (see Section 4.2.15)	F	G, H
Organizational Model for Enterprise Architecture (see Section 4.2.16)	Preliminary	Preliminary, A, B, C, D, E, F, G, H, Requirements Management
Request for Architecture Work (see Section 4.2.17)	Preliminary, F, H	A, G
Requirements Impact Assessment (see Section 4.2.18)	Requirements Management	Requirements Management
Solution Building Blocks (see Section 4.2.19)	G	A, B, C, D, E, F, G
Statement of Architecture Work (see Section 4.2.20)	A, B, C, D, E, F, G, H	B, C, D, E, F, G, H, Requirements Management
Tailored Architecture Framework (see Section 4.2.21)	Preliminary, A	Preliminary, A, B, C, D, E, F, G, H, Requirements Management

4.2 Deliverable Descriptions

The following sections provide example descriptions of deliverables referenced in the ADM.

Note that not all the content described here need be contained in a particular deliverable. Rather, it is recommended that external references be used where possible; for example, the strategic plans of a business should not be copied into a Request for Architecture Work, but rather the title of the strategic plans should be referenced.

Also, it is not suggested that these descriptions should be followed to the letter. However, each element should be considered carefully; ignoring any input or output item may cause problems downstream.

4.2.1 Architecture Building Blocks

Architecture documentation and models from the enterprise's Architecture Repository; see Chapter 5.

4.2.2 Architecture Contract

Purpose

Architecture Contracts are the joint agreements between development partners and sponsors on the deliverables, quality, and fitness-for-purpose of an architecture. Successful implementation of these agreements will be delivered through effective Architecture Governance (see the TOGAF Standard — Enterprise Architecture Capability and Governance). By implementing a governed approach to the management of contracts, the following will be ensured:

- A system of continuous monitoring to check integrity, changes, decision-making, and audit of all architecture-related activities within the organization
- Adherence to the principles, standards, and requirements of the existing or developing architectures
- Identification of risks in all aspects of the development and implementation of the architecture(s) covering the internal development against accepted standards, policies, technologies, and products as well as the operational aspects of the architectures such that the organization can continue its business within a resilient environment
- A set of processes and practices that ensure accountability, responsibility, and discipline with regard to the development and usage of all architectural artifacts
- A formal understanding of the governance organization responsible for the contract, their level of authority, and scope of the architecture under the governance of this body

Content

Typical contents of an Architecture Design and Development Contract are:

- Introduction and background
- The nature of the agreement
- Scope of the architecture
- Architecture and strategic principles and requirements
- Conformance requirements
- Architecture development and management process and roles
- Target Architecture measures
- Defined phases of deliverables
- Prioritized joint workplan
- Time window(s)
- Architecture delivery and business metrics

Typical contents of a Business Users' Architecture Contract are:

- Introduction and background
- The nature of the agreement
- Scope
- Strategic requirements
- Conformance requirements
- Architecture adopters
- Time window
- Architecture business metrics
- Service architecture (includes Service-Level Agreement (SLA))

For more detail on the use of Architecture Contracts, see the TOGAF Standard — Enterprise Architecture Capability and Governance.

4.2.3 Architecture Definition Document

Purpose

The Architecture Definition Document is the deliverable container for the core architectural artifacts created during a project and for important related information. The Architecture Definition Document spans all architecture domains (Business, Data, Application, and Technology) and also examines all relevant states of the architecture (Baseline, Transition, and Target).

A Transition Architecture shows the enterprise at an architecturally significant state between the Baseline and Target Architectures. Transition Architectures are used to describe transitional Target Architectures necessary for effective realization of the Target Architecture.

The Architecture Definition Document is a companion to the Architecture Requirements Specification, with a complementary objective:

- The Architecture Definition Document provides a qualitative view of the solution and aims to communicate the intent of the architects
- The Architecture Requirements Specification provides a quantitative view of the solution, stating measurable criteria that must be met during the implementation of the architecture

Content

Typical contents of an Architecture Definition Document are:

- Scope
- Goals, objectives, and constraints
- Architecture Principles
- Baseline Architecture

- Architecture models (for each state to be modeled):
  - Business Architecture models
  - Data Architecture models
  - Application Architecture models
  - Technology Architecture models
- Rationale and justification for architectural approach
- Mapping to Architecture Repository:
  - Mapping to Architecture Landscape
  - Mapping to reference models
  - Mapping to standards
  - Re-use assessment
- Gap analysis
- Impact assessment
- Transition Architecture:
  - Definition of transition states
  - Business Architecture for each transition state
  - Data Architecture for each transition state
  - Application Architecture for each transition state
  - Technology Architecture for each transition state

4.2.4 Architecture Principles

Purpose

Principles are general rules and guidelines, intended to be enduring and seldom amended, that inform and support the way in which an organization sets about fulfilling its mission.

In their turn, principles may be just one element in a structured set of ideas that collectively define and guide the organization, from values through to actions and results.

Content

See the TOGAF Standard — ADM Techniques for guidelines and a detailed set of generic Architecture Principles, including:

- Business principles (see the TOGAF Standard — ADM Techniques)
- Data principles (see the TOGAF Standard — ADM Techniques)
- Application principles (see the TOGAF Standard — ADM Techniques)
- Technology principles (see the TOGAF Standard — ADM Techniques)

4.2.5 Architecture Repository

Purpose

The Architecture Repository acts as a holding area for all architecture-related projects within the enterprise. The repository allows projects to manage their deliverables, locate re-usable assets, and publish outputs to stakeholders and other interested parties.

Content

See Chapter 7 for a detailed description of the content of an Architecture Repository.

4.2.6 Architecture Requirements Specification

Purpose

The Architecture Requirements Specification provides a set of quantitative statements that outline what an implementation project must do in order to comply with the architecture. An Architecture Requirements Specification will typically form a major component of an implementation contract or contract for more detailed Architecture Definition.

As mentioned above, the Architecture Requirements Specification is a companion to the Architecture Definition Document, with a complementary objective:

- The Architecture Definition Document provides a qualitative view of the solution and aims to communicate the intent of the architect
- The Architecture Requirements Specification provides a quantitative view of the solution, stating measurable criteria that must be met during the implementation of the architecture

Content

Typical contents of an Architecture Requirements Specification are:

- Success measures
- Architecture requirements
- Business service contracts
- Application service contracts
- Implementation guidelines
- Implementation specifications
- Implementation standards
- Interoperability requirements
- IT Service Management requirements
- Constraints
- Assumptions

4.2.7 Architecture Roadmap

Purpose

The Architecture Roadmap lists individual work packages that will realize the Target Architecture and lays them out on a timeline to show progression from the Baseline Architecture to the Target Architecture. The Architecture Roadmap highlights individual work packages' business value at each stage. Transition Architectures necessary to effectively realize the Target Architecture are identified as intermediate steps. The Architecture Roadmap is incrementally developed throughout Phases E and F, and informed by readily identifiable roadmap components from Phase B, C, and D within the ADM.

Content

Typical contents of an Architecture Roadmap are:

- Work package portfolio:
  - Work package description (name, description, objectives, deliverables)
  - Functional requirements
  - Dependencies
  - Relationship to opportunity
  - Relationship to Architecture Definition Document and Architecture Requirements Specification
  - Business value
- Implementation Factor catalog, including:
  - Risks
  - Issues
  - Assumptions
  - Dependencies
  - Actions
  - Inputs
- Consolidated Gaps, Solutions, and Dependencies matrix, including:
  - Architecture domain
  - Gap
  - Potential solutions
  - Dependencies
- Any Transition Architectures
- Implementation recommendations:
  - Criteria measures of effectiveness of projects

- — Risks and issues
- — Solution Building Blocks

4.2.8 Architecture Vision

Purpose

The Architecture Vision is created early on in the ADM cycle. It provides a summary of the changes to the enterprise that will accrue from successful deployment of the Target Architecture. The purpose of the Architecture Vision is to provide key stakeholders with a formally agreed outcome. Early agreement on the outcome enables the architects to focus on the detail necessary to validate feasibility. Providing an Architecture Vision also supports stakeholder communication by providing a summary version of the full Architecture Definition.

Content

Typical contents of an Architecture Vision are:

- Problem description:
  - — Stakeholders and their concerns
  - — List of issues/scenarios to be addressed
- Objective of the Statement of Architecture Work
- Summary views necessary for the Request for Architecture Work and the Draft Business, Data, Application, and Technology Architectures created; typically including:
  - — Value Chain diagram
  - — Solution Concept diagram
- Mapped requirements
- Reference to Draft Architecture Definition Document

4.2.9 Business Principles, Business Goals, and Business Drivers

Purpose

Business principles, business goals, and business drivers provide context for architecture work, by describing the needs and ways of working employed by the enterprise. Many factors that lie outside the consideration of architecture discipline may nevertheless have significant implications for the way that architecture is developed.

Content

The content and structure of business context for architecture is likely to vary considerably from one organization to the next.

4.2.10 Capability Assessment

Purpose

Before embarking upon a detailed Architecture Definition, it is valuable to understand the baseline and target capability level of the enterprise. This Capability Assessment can be examined on several levels:

- What is the capability level of the enterprise as a whole? Where does the enterprise wish to increase or optimize capability? What are the architectural focus areas that will support the desired development of the enterprise?
- What is the capability or maturity level of the IT function within the enterprise? What are the likely implications of conducting the architecture project in terms of design governance, operational governance, skills, and organization structure? What is an appropriate style, level of formality, and amount of detail for the architecture project to fit with the culture and capability of the IT organization?
- What is the capability and maturity of the architecture function within the enterprise? What architectural assets are currently in existence? Are they maintained and accurate? What standards and reference models need to be considered? Are there likely to be opportunities to create re-usable assets during the architecture project?
- Where capability gaps exist, to what extent is the business ready to transform in order to reach the target capability? What are the risks to transformation, cultural barriers, and other considerations to be addressed beyond the basic capability gap?

Content

Typical contents of a Capability Assessment are:

- Business Capability Assessment, including:
  - Capabilities of the business
  - Baseline state assessment of the performance level of each capability
  - Future state aspiration for the performance level of each capability
  - Baseline state assessment of how each capability is realized
  - Future state aspiration for how each capability should be realized
  - Assessment of likely impacts to the business organization resulting from the successful deployment of the Target Architecture
- IT Capability Assessment, including:
  - Baseline and target maturity level of change process
  - Baseline and target maturity level of operational processes
  - Baseline capability and capacity assessment
  - Assessment of the likely impacts to the IT organization resulting from the successful deployment of the Target Architecture

- Architecture maturity assessment, including:
  - Architecture Governance processes, organization, roles, and responsibilities
  - Architecture skills assessment
  - Breadth, depth, and quality of landscape definition with the Architecture Repository
  - Breadth, depth, and quality of standards definition with the Architecture Repository
  - Breadth, depth, and quality of reference model definition with the Architecture Repository
  - Assessment of re-use potential
- Business Transformation Readiness Assessment, including:
  - Readiness factors
  - Vision for each readiness factor
  - Current and target readiness ratings
  - Readiness risks

4.2.11 Change Request

Purpose

During implementation of an architecture, as more facts become known, it is possible that the original Architecture Definition and requirements are not suitable or are not sufficient to complete the implementation of a solution. In these circumstances, it is necessary for implementation projects to either deviate from the suggested architectural approach or to request scope extensions. Additionally, external factors — such as market factors, changes in business strategy, and new technology opportunities — may open up opportunities to extend and refine the architecture.

In these circumstances, a Change Request may be submitted in order to kick-start a further cycle of architecture work.

Content

Typical contents of a Change Request are:

- Description of the proposed change
- Rationale for the proposed change
- Impact assessment of the proposed change, including:
  - Reference to specific requirements
  - Stakeholder priority of the requirements to date
  - Phases to be revisited
  - Phase to lead on requirements prioritization
  - Results of phase investigations and revised priorities

  - Recommendations on management of requirements
- Repository reference number

4.2.12 Communications Plan

Purpose

Enterprise Architectures contain large volumes of complex and inter-dependent information. Effective communication of targeted information to the right stakeholders at the right time is a CSF for Enterprise Architecture. Development of a Communications Plan for architecture allows for this communication to be carried out within a planned and managed process.

Content

Typical contents of a Communications Plan are:

- Identification of stakeholders and grouping by communication requirements
- Identification of communication needs, key messages in relation to the Architecture Vision, communication risks, and CSFs
- Identification of mechanisms that will be used to communicate with stakeholders and allow access to architecture information, such as meetings, newsletters, repositories, etc.
- Identification of a communications timetable, showing which communications will occur with which stakeholder groups at what time and in what location

4.2.13 Compliance Assessment

Purpose

Once an architecture has been defined, it is necessary to govern that architecture through implementation to ensure that the original Architecture Vision is appropriately realized and that any implementation learnings are fed back into the architecture process. Periodic compliance reviews of implementation projects provide a mechanism to review project progress and ensure that the design and implementation is proceeding in line with the strategic and architectural objectives.

Content

Typical contents of a Compliance Assessment are:

- Overview of project progress and status
- Overview of project architecture/design
- Completed architecture checklists:
  - Hardware and operating system checklist
  - Software services and middleware checklist
  - Applications checklists

- Information management checklists
- Security checklists
- System management checklists
- System engineering checklists
- Methods and tools checklists

4.2.14 Implementation and Migration Plan

Purpose

The Implementation and Migration Plan provides a schedule of the projects that will realize the Target Architecture. The Implementation and Migration Plan includes executable projects grouped into managed portfolios and programs. The Implementation and Migration Strategy identifying the approach to change is a key element of the Implementation and Migration Plan.

Content

Typical contents of an Implementation and Migration Plan are:

- Implementation and Migration Strategy:
  - Strategic implementation direction
  - Implementation sequencing approach
- Project and portfolio breakdown of implementation:
  - Allocation of work packages to project and portfolio
  - Capabilities delivered by projects
  - Milestones and timing
  - Work breakdown structure
  - May include impact on existing portfolio, program, and projects

It may contain:

- Project charters:
  - Included work packages
  - Business value
  - Risk, issues, assumptions, dependencies
  - Resource requirements and costs
  - Benefits of migration, determined (including mapping to business requirements)
  - Estimated costs of migration options

4.2.15 Implementation Governance Model

Purpose

Once an architecture has been defined, it is necessary to plan how the Transition Architecture that implements the architecture will be governed through implementation. Within organizations that have established architecture functions, there is likely to be a governance framework already in place, but specific processes, organizations, roles, responsibilities, and measures may need to be defined on a project-by-project basis.

The Implementation Governance Model ensures that a project transitioning into implementation also smoothly transitions into appropriate Architecture Governance.

Content

Typical contents of an Implementation Governance Model are:

- Governance processes
- Governance organization structure
- Governance roles and responsibilities
- Governance checkpoints and success/failure criteria

4.2.16 Organizational Model for Enterprise Architecture

Purpose

In order for an architecture framework to be used successfully, it must be supported by the correct organization, roles, and responsibilities within the enterprise. Of particular importance is the definition of boundaries between different Enterprise Architecture practitioners and the governance relationships that span across these boundaries.

Content

Typical contents of an Organizational Model for Enterprise Architecture are:

- Scope of organizations impacted
- Maturity assessment, gaps, and resolution approach
- Roles and responsibilities for architecture team(s)
- Constraints on architecture work
- Budget requirements
- Governance and support strategy

4.2.17 Request for Architecture Work

Purpose

This is a document that is sent from the sponsoring organization to the architecture organization to trigger the start of an architecture development cycle. Requests for Architecture Work can be created as an output of the Preliminary Phase, a result of approved architecture Change Requests, or terms of reference for architecture work originating from migration planning.

In general, all the information in this document should be at a high level.

Content

Requests for Architecture Work typically include:

- Organization sponsors
- Organization's mission statement
- Business goals (and changes)
- Strategic plans of the business
- Time limits
- Changes in the business environment
- Organizational constraints
- Budget information, financial constraints
- External constraints, business constraints
- Current business system description
- Current architecture/IT system description
- Description of developing organization
- Description of resources available to developing organization

4.2.18 Requirements Impact Assessment

Purpose

Throughout the ADM, new information is collected relating to an architecture. As this information is gathered, new facts may come to light that invalidate existing aspects of the architecture. A Requirements Impact Assessment assesses the current architecture requirements and specification to identify changes that should be made and the implications of those changes.

Content

Typical contents of a Requirements Impact Assessment are:

- Reference to specific requirements
- Stakeholder priority of the requirements to date
- Phases to be revisited

- Phase to lead on requirements prioritization
- Results of phase investigations and revised priorities
- Recommendations on management of requirements
- Repository reference number

4.2.19 Solution Building Blocks

Implementation-specific building blocks from the enterprise's Architecture Repository; see Chapter 5.

4.2.20 Statement of Architecture Work

Purpose

The Statement of Architecture Work defines the scope and approach that will be used to complete an architecture development cycle. The Statement of Architecture Work is typically the document against which successful execution of the architecture project will be measured and may form the basis for a contractual agreement between the supplier and consumer of architecture services.

Content

Typical contents of a Statement of Architecture Work are:

- Title
- Architecture project request and background
- Architecture project description and scope
- Overview of Architecture Vision
- Specific change of scope procedures
- Roles, responsibilities, and deliverables
- Acceptance criteria and procedures
- Architecture project plan and schedule
- Approvals

4.2.21 Tailored Architecture Framework

Purpose

The TOGAF framework provides an industry standard for architecture that may be used in a wide variety of organizations. However, before the TOGAF framework can be effectively used within an architecture project, tailoring at two levels is necessary.

Firstly, it is necessary to tailor the TOGAF model for integration into the enterprise. This tailoring will include integration with management frameworks, customization of terminology, development of presentational styles, selection, configuration, and deployment of architecture tools, etc. The formality and detail of any frameworks adopted should also align with other contextual factors for the enterprise, such as culture, stakeholders, commercial models for Enterprise Architecture, and the existing level of Architecture Capability.

Once the framework has been tailored to the enterprise, further tailoring is necessary in order to tailor the framework for the specific architecture project. Tailoring at this level will select appropriate deliverables and artifacts to meet project and stakeholder needs.

See the TOGAF Standard — Architecture Development Method for further considerations when selecting and tailoring the architecture framework.

Content

Typical contents of a Tailored Architecture Framework are:

- Tailored architecture method
- Tailored architecture content (deliverables and artifacts)
- Configured and deployed tools
- Interfaces with governance models and other frameworks:
  - Corporate Business Planning
  - Enterprise Architecture
  - Portfolio, Program, Project Management
  - System Development/Engineering
  - Operations (Services)

Chapter 5: Building Blocks

This chapter explains the concept of building blocks.

5.1 Overview

This section is intended to explain and illustrate the concept of building blocks in architecture.

Following this overview, there are two main parts:

- Introduction to Building Blocks (see Section 5.2), discusses the general concepts of building blocks, and explains the differences between ABBs and SBBs
- Building Blocks and the ADM (see Section 5.3), summarizes the stages at which building block design and specification occurs within the TOGAF ADM

5.2 Introduction to Building Blocks

This section is an introduction to the concept of building blocks.

5.2.1 Overview

This section describes the characteristics of building blocks. The use of building blocks in the ADM is described separately in Section 5.3.

5.2.2 Generic Characteristics

Building blocks have generic characteristics as follows:

- A building block is a package of functionality defined to meet the business needs across an organization
- A building block normally has a type that corresponds to the metamodel (such as actor, business service, application, or data entity)
- A building block has a defined boundary and is generally recognizable as "a thing" by domain experts
- A building block may interoperate with other, inter-dependent building blocks.
- A good building block has the following characteristics:
  - It considers implementation and usage, and evolves to exploit technology and standards

— It may be assembled from other building blocks

— It may be a subassembly of other building blocks

— Ideally a building block is re-usable and replaceable, and well specified

A building block's boundary and specification should be loosely coupled to its implementation; i.e., it should be possible to realize a building block in several different ways without impacting the boundary or specification of the building block. The way in which assets and capabilities are assembled into building blocks will vary widely between individual architectures. Every organization must decide for itself what arrangement of building blocks works best for it. A good choice of building blocks can lead to improvements in legacy system integration, interoperability, and flexibility in the creation of new systems and applications.

Systems are built up from collections of building blocks, so most building blocks have to interoperate with other building blocks. Wherever that is true, it is important that the interfaces to a building block are published and reasonably stable.

Building blocks can be defined at various levels of detail, depending on what stage of architecture development has been reached.

For instance, at an early stage, a building block can simply consist of a name or an outline description. Later on, a building block may be decomposed into multiple supporting building blocks and may be accompanied by a full specification.

The level of detail to which a building block should be specified is dependent on the objectives of the architecture and, in some cases, less detail may be of greater value (for example, when presenting the capabilities of an enterprise, a single clear and concise picture has more value than a dense 100-page specification).

The Object Management Group® (OMG®) has developed a standard for Re-usable Asset Specification (RAS),[1] which provides a good example of how building blocks can be formally described and managed.

5.2.3 Architecture Building Blocks

Architecture Building Blocks (ABBs) relate to the Architecture Continuum (see Section 6.4.1), and are defined or selected as a result of the application of the ADM.

5.2.3.1 Characteristics

ABBs:

- Capture architecture requirements; e.g., Business, Data, Application, and Technology requirements
- Direct and guide the development of SBBs

1. Refer to www.omg.org/spec/RAS.

5.2.3.2 Specification Content

ABB specifications include the following as a minimum:

- Fundamental functionality and attributes: semantic, unambiguous, including security capability and manageability
- Interfaces: chosen set, supplied
- Interoperability and relationship with other building blocks
- Dependent building blocks with required functionality and named user interfaces
- Map to business/organizational entities and policies

5.2.4 Solution Building Blocks

Solution Building Blocks (SBBs) relate to the Solutions Continuum (see Section 6.4.2), and may be either procured or developed.

5.2.4.1 Characteristics

SBBs:

- Define what products and components will implement the functionality
- Define the implementation
- Fulfil business requirements
- Are product or vendor-aware

5.2.4.2 Specification Content

SBB specifications include the following as a minimum:

- Specific functionality and attributes
- Interfaces; the implemented set
- Required SBBs used with required functionality and names of the interfaces used
- Mapping from the SBBs to the IT topology and operational policies
- Specifications of attributes shared across the environment (not to be confused with functionality) such as security, manageability, localizability, scalability
- Performance, configurability
- Design drivers and constraints, including the physical architecture
- Relationships between SBBs and ABBs

5.3 Building Blocks and the ADM

5.3.1 Basic Principles

This section focuses on the use of building blocks in the ADM. General considerations and characteristics of building blocks are described in Section 5.2.

5.3.1.1 Building Blocks in Architecture Design

An architecture is a set of building blocks depicted in an architectural model, and a specification of how those building blocks are connected to meet the overall requirements of the business.

The various building blocks in an architecture specify the scope and approach that will be used to address a specific business problem.

There are some general principles underlying the use of building blocks in the design of specific architectures:

- An architecture need only contain building blocks that are relevant to the business problem that the architecture is attempting to address
- Building blocks may have complex relationships to one another

  One building block may support multiple building blocks or may partially support a single building block (for example, the business service of "complaint handling" would be supported by many data entities and possibly multiple application components)
- Building blocks should conform to standards relevant to their type, the principles of the enterprise, and the standards of the enterprise

5.3.1.2 Building Block Design

The process of identifying building blocks includes looking for collections of capabilities or assets that interact with one another and then drawing them together or making them different:

- Consider three classes of building blocks:
  - Re-usable building blocks, such as legacy items
  - Building blocks to be the subject of development, such as new applications
  - Building blocks to be the subject of purchase; i.e., Commercial Off-The-Shelf (COTS) applications
- Use the desired level of integration to bind or combine functions into building blocks; for instance, legacy elements could be treated as large building blocks to avoid breaking them apart

In the early stages and during views of the highest-level enterprise, the building blocks are often kept at a broad integration definition. It is during these exercises that the services definitions can often be best viewed. As implementation considerations are addressed, more detailed views of building blocks can often be used to address implementation decisions, focus on the critical strategic decisions, or aid in assessing the value and future impact of commonality and re-usability.

5.3.2 Building Block Specification Process in the ADM

The process of building block definition takes place gradually as the ADM is followed, mainly in Phases A, B, C, and D. It is an evolutionary and iterative process because as definition proceeds, detailed information about the functionality required, the constraints imposed on the architecture, and the availability of products may affect the choice and the content of building blocks.

The key phases and steps of the ADM at which building blocks are evolved and specified are summarized in Figure 5-1. The major work in these steps consists of identifying the ABBs required to meet the business goals and objectives. The selected set of ABBs is then refined in an iterative process to arrive at a set of SBBs which can either be bought off-the-shelf or custom developed.

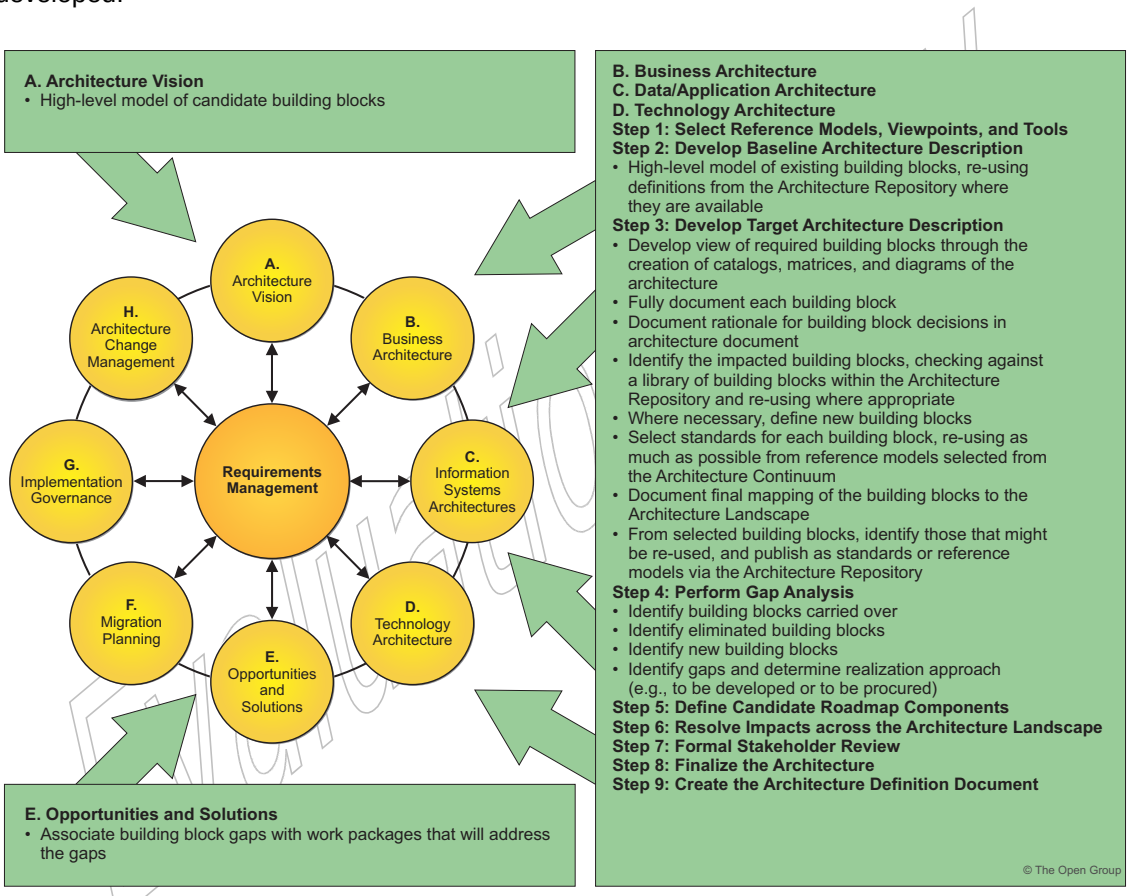

Figure 5-1 Key ADM Phases/Steps at which Building Blocks are Evolved/Specified

Chapter 6: Enterprise Continuum

6.1 Overview

The Enterprise Continuum provides methods for classifying architecture and solution artifacts, both internal and external to the Architecture Repository, as they evolve from generic Foundation Architectures to Organization-Specific Architectures.

The Enterprise Continuum enables the architect to articulate the broad perspective of what, why, and how the Enterprise Architecture has been designed with the factors and drivers considered. The Enterprise Continuum is an important aid to communication and understanding, both within individual enterprises, and between customer enterprises and vendor organizations. Without an understanding of "where in the continuum you are", people discussing architecture can often talk at cross-purposes because they are referencing different points in the continuum at the same time, without realizing it.

Any architecture is context-specific; for example, there are architectures that are specific to individual customers, industries, subsystems, products, and services. Architects, on both the buy side and supply side, must have at their disposal a consistent language to effectively communicate the differences between architectures. Such a language will enable engineering efficiency and the effective leveraging of COTS product functionality. The Enterprise Continuum provides that consistent language.

The Enterprise Continuum enables the organization of re-usable architecture artifacts and solution assets to maximize the Enterprise Architecture investment opportunities.

6.2 Enterprise Continuum and Architecture Re-Use

The simplest way of thinking of the Enterprise Continuum is as a view of the repository of all the architecture assets. It can contain Architecture Descriptions, models, building blocks, patterns, architecture viewpoints, and other artifacts — that exist both within the enterprise and in the IT industry at large, which the enterprise considers to have available for the development of architectures for the enterprise.

Examples of internal architecture and solution artifacts are the deliverables of previous architecture work, which are available for re-use. Examples of external architecture and solution artifacts are the wide variety of industry reference models and architecture patterns that exist, and are continually emerging, including those that are highly generic (such as the TOGAF TRM); those specific to certain aspects of IT (such as a web services architecture, or a generic manageability architecture); those specific to certain types of information processing, such as e-Commerce, supply chain management, etc.; and those specific to certain vertical industries, such as the models generated by vertical consortia like the TM Forum (in the Telecommunications sector), ARTS (Retail), Energistics® (Petrotechnical), etc.

The Enterprise Architecture determines which architecture and solution artifacts an organization includes in its Architecture Repository. Re-use is a major consideration in this decision.

6.3 Constituents of the Enterprise Continuum

An overview of the context and constituents of the Enterprise Continuum is shown in Figure 6-1.

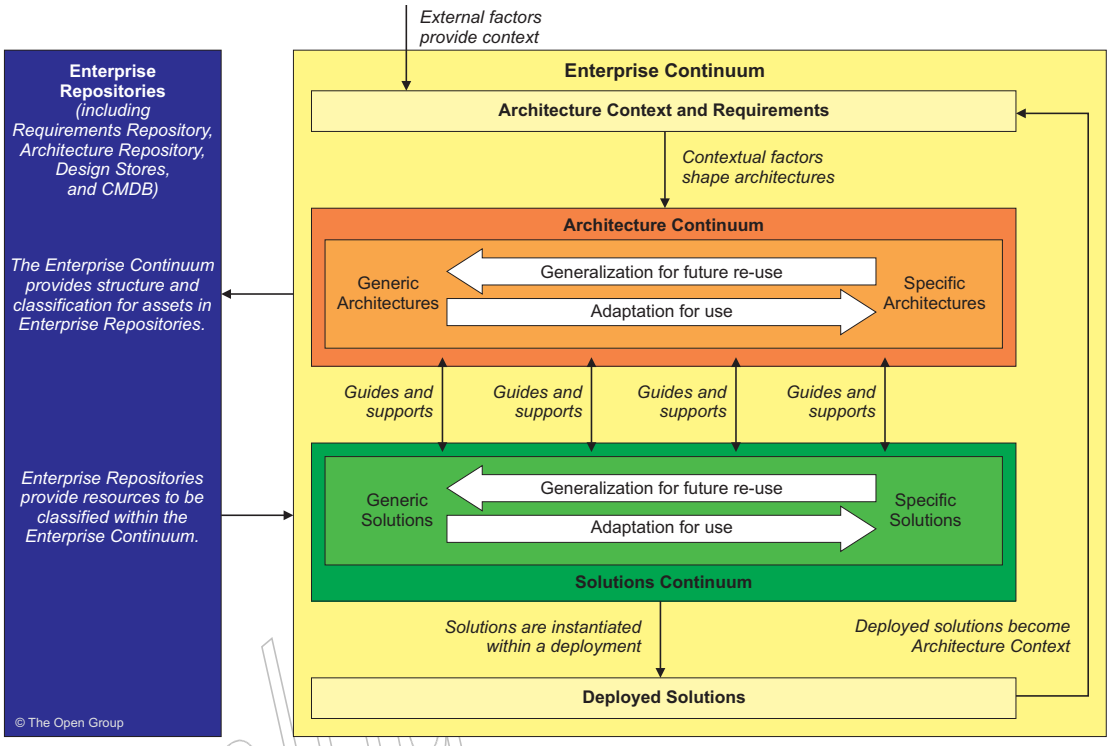

Figure 6-1 Enterprise Continuum

The Enterprise Continuum is partitioned into three distinct continua as follows:

- The **Enterprise Continuum** (see Section 6.4) is the outermost continuum and classifies assets related to the context of the overall Enterprise Architecture

  The Enterprise Continuum classes of assets may influence architectures, but are not directly used during the ADM architecture development. The Enterprise Continuum classifies contextual assets used to develop architectures, such as policies, standards, strategic initiatives, organizational structures, and enterprise-level capabilities. The Enterprise Continuum can also classify solutions (as opposed to descriptions or specifications of solutions). Finally, the Enterprise Continuum contains two specializations, namely the Architecture and Solutions Continua.

- The **Architecture Continuum** (see Section 6.4.1) offers a consistent way to define and understand the generic rules, representations, and relationships in an architecture, including traceability and derivation relationships (e.g., to show that an Organization-Specific Architecture is based on an industry or generic standard)

The Architecture Continuum represents a structuring of Architecture Building Blocks (ABBs) which are re-usable architecture assets. ABBs evolve through their development lifecycle from abstract and generic entities to fully expressed Organization-Specific Architecture assets. The Architecture Continuum assets will be used to guide and select the elements in the Solutions Continuum (see below). The Architecture Continuum shows the relationships among foundational frameworks (such as the TOGAF framework), common system architectures (such as the III-RM), industry architectures, and Enterprise Architectures. The Architecture Continuum is a useful tool to discover commonality and eliminate unnecessary redundancy.

- The **Solutions Continuum** (see Section 6.4.2) provides a consistent way to describe and understand the implementation of the assets defined in the Architecture Continuum

  The Solutions Continuum defines what is available in the organizational environment as re-usable SBBs. The solutions are the results of agreements between customers and business partners that implement the rules and relationships defined in the architecture space. The Solutions Continuum addresses the commonalities and differences among the products, systems, and services of implemented systems.

The Enterprise Continuum classifies architecture assets that are applicable across the entire scope of the Enterprise Architecture. These assets, which may be referred to as building blocks, can represent a variety of elements that collectively define and constrain the Enterprise Architecture. They can take the form of business goals and objectives, strategic initiatives, capabilities, policies, standards, and principles.

The Enterprise Continuum also contains the Architecture Continuum and the Solutions Continuum. Each of these continua is described in greater detail in the following sections.

6.4 Enterprise Continuum in Detail

The Enterprise Continuum is intended to represent the classification of all assets that are available to an enterprise. It classifies assets that exist within the enterprise along with other assets in the wider environment that are relevant to the enterprise, such as products, research, market factors, commercial factors, business strategies, and legislation.

The TOGAF Standard is intended to be a framework for conducting Enterprise Architecture and as a result many of the assets that reside within the Enterprise Continuum are beyond the specific consideration of the TOGAF framework. However, architectures are fundamentally shaped by concerns outside the practice of architecture and it is therefore of paramount importance that any architecture must accurately reflect external context.

The specific contextual factors to be identified and incorporated in an architecture will vary from architecture to architecture. However, typical contextual factors for architecture development are likely to include:

- External influencing factors, such as regulatory change, technological advances, and competitor activity
- Business strategy and context, including mergers, acquisitions, and other business transformation requirements
- Current business operations, reflecting deployed architectures and solutions

By observing the context for architecture, it can be seen that architecture development activity exists within a wider enterprise lifecycle of continuous change.

ABBs are defined in relation to a set of contextual factors and then realized through SBBs. SBBs are deployed as live solutions and become a part of the baseline operating model of the enterprise. The operating model of the enterprise and empiric information on the performance of the enterprise shapes the context and requirements for future change. Finally, these new requirements for change create a feedback loop to influence the creation of new Target Architectures.

6.4.1 Architecture Continuum

The Architecture Continuum illustrates how architectures are developed and evolved across a continuum ranging from Foundation Architectures, such as the TOGAF® Series Guide: The TOGAF® Technical Reference Model (TRM) through Common Systems Architectures, and Industry Architectures, and to an enterprise's own Organization-Specific Architectures.

The arrows in the Architecture Continuum represent the relationship that exists between the different architectures in the Architecture Continuum. The leftwards direction focuses on meeting enterprise needs and business requirements, while the rightwards direction focuses on leveraging architectural components and building blocks.

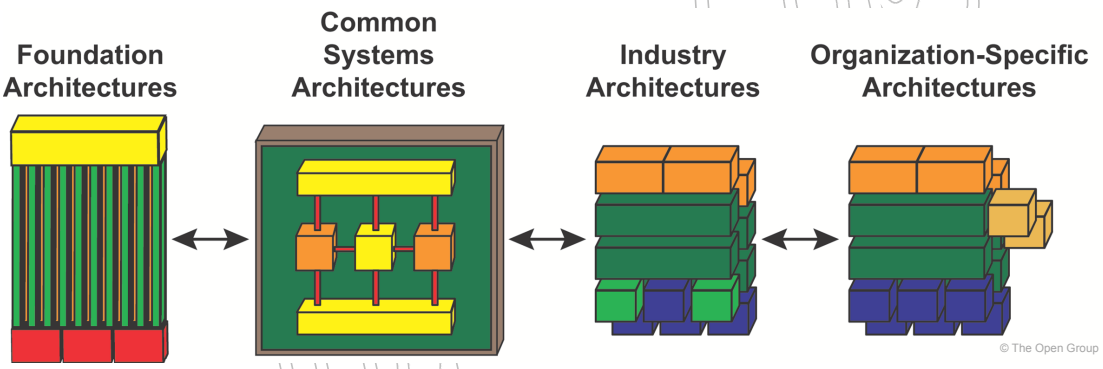

Figure 6-2 Architecture Continuum

The enterprise needs and business requirements are addressed in increasing detail from left to right. The architect will typically look to find re-usable architectural elements toward the left of the continuum. When elements are not found, the requirements for the missing elements are passed to the left of the continuum for incorporation. Those implementing architectures within their own organizations can use the same continuum models specialized for their business.

The four particular architecture types illustrated in Figure 6-2 are intended to indicate the range of different types of architecture that may be developed at different points in the continuum; they are not fixed stages in a process.

Many different types of architecture may occur at points in between those illustrated in Figure 6-2. Although the evolutionary transformation continuum illustrated does not represent a formal process, it does represent a progression, which occurs at several levels:

- Logical to physical

- Horizontal (IT-focused) to vertical (business-focused)
- Generalization to specialization
- Taxonomy to complete and specific architecture specification

At each point in the continuum, an architecture is designed in terms of the design concepts and building blocks available and relevant to that point.

The four architectures illustrated in Figure 6-2 represent main classifications of potential architectures, and will be relevant and familiar to many architects. They are analyzed in detail below.

Foundation Architecture

A Foundation Architecture consists of generic components, inter-relationships, principles, and guidelines that provide a foundation on which more specific architectures can be built. The TOGAF ADM is a process that would support specialization of such Foundation Architectures in order to create organization-specific models.

The TOGAF TRM is an example of a Foundation Architecture. It is a fundamental architecture upon which other, more specific architectures can be based. See the TOGAF® Series Guide: The TOGAF® Technical Reference Model (TRM) for more details.

Common Systems Architectures

Common Systems Architectures guide the selection and integration of specific services from the Foundation Architecture to create an architecture useful for building common (i.e., highly re-usable) solutions across a wide number of relevant domains.

Examples of Common Systems Architectures include: a security architecture, a management architecture, a network architecture, an operations architecture, etc. Each is incomplete in terms of overall system functionality, but is complete in terms of a particular problem domain (security, manageability, networking, operations, etc.), so that solutions implementing the architecture constitute re-usable building blocks for the creation of functionally complete operating states of the enterprise.

Other characteristics of Common Systems Architectures include:

- Reflects requirements specific to a generic problem domain
- Defines building blocks specific to a generic problem domain
- Defines business, data, application, or technology standards for implementing these building blocks
- Provides building blocks for easy re-use and lower costs

The TOGAF Integrated Information Infrastructure Reference Model (III-RM) — see the TOGAF® Series Guide: The TOGAF Integrated Information Infrastructure Reference Model (III-RM) — is a reference model that supports describing Common Systems Architecture in the Application Domain that focuses on the requirements, building blocks, and standards relating to the vision of Boundaryless Information Flow.

Industry Architectures

Industry Architectures guide the integration of common systems components with industry-specific components, and guide the creation of industry solutions for targeted customer problems within a particular industry.

A typical example of an industry-specific component is a data model representing the business capabilities and processes specific to a particular vertical industry, such as the Retail industry's "Active Store" architecture, or an Industry Architecture that incorporates the Energistics Data Model (refer to www.energistics.org).

Other characteristics of Industry Architectures include:

- Reflects requirements and standards specific to a vertical industry
- Defines building blocks specific to a generic problem domain
- Contains industry-specific logical data and process models
- Contains industry-specific applications and process models, as well as industry-specific business rules
- Provides guidelines for testing collections of systems
- Encourages levels of interoperability throughout the industry

Organization-Specific Architectures

Organization-Specific Architectures describe and guide the final deployment of solution components for a particular enterprise or extended network of connected enterprises.

There may be a variety of Organization-Specific Architectures that are needed to effectively cover the organization's requirements by defining the architectures in increasing levels of detail. Alternatively, this might result in several more detailed Organization-Specific Architectures for specific entities within the global enterprise. Breaking down Organization-Specific Architectures into constituent pieces is addressed in the TOGAF Standard — Applying the ADM.

The Organization-Specific Architecture guides the final customization of the solution, and has the following characteristics:

- Provides a means to communicate and manage business operations across all four architecture domains
- Reflects requirements specific to a particular enterprise
- Defines building blocks specific to a particular enterprise
- Contains organization-specific business models, data, applications, and technologies
- Provides a means to encourage implementation of appropriate solutions to meet business needs
- Provides the criteria to measure and select appropriate products, solutions, and services
- Provides an evolutionary path to support growth and new business needs

6.4.2 Solutions Continuum

The Solutions Continuum represents the detailed specification and construction of the architectures at the corresponding levels of the Architecture Continuum. At each level, the Solutions Continuum is a population of the architecture with reference building blocks — either purchased products or built components — that represent a solution to the enterprise's business need expressed at that level. A populated repository based on the Solutions Continuum can be regarded as a solutions inventory or re-use library, which can add significant value to the task of managing and implementing improvements to the enterprise.

The Solutions Continuum is illustrated in Figure 6-3.

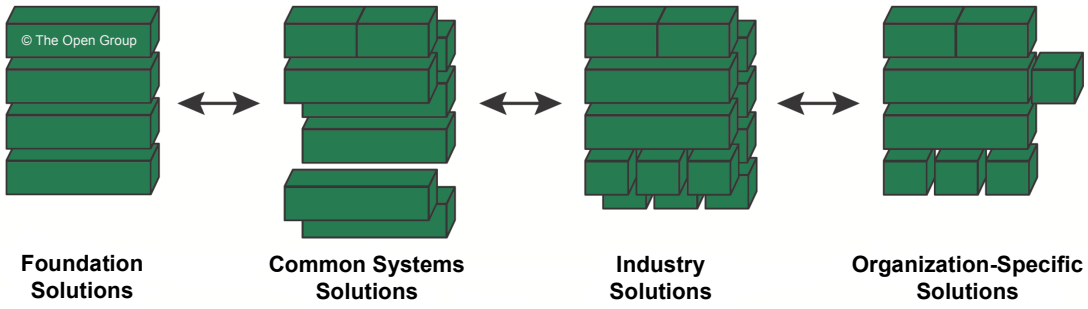

Figure 6-3 Solutions Continuum

"Moving to the right" on the Solutions Continuum is focused on providing solutions value (i.e., foundation solutions provide value in creating common systems solutions; common systems solutions are used to create industry solutions; and industry solutions are used to create organization-specific solutions). "Moving to the left" on the Solutions Continuum is focused on addressing enterprise needs. These two viewpoints are significant for a company attempting to focus on its needs while maximizing the use of available resources through leverage.

The following subsections describe each of the solution types within the Solutions Continuum.

Foundation Solutions

Foundation Solutions are highly generic concepts, tools, products, services, and solution components that are the fundamental providers of capabilities. Services include professional services — such as training and consulting services — that ensure the maximum investment value from solutions in the shortest possible time; and support services — such as Help Desk — that ensure the maximum possible value from solutions (services that ensure timely updates and upgrades to the products and systems).

Example Foundation Solutions would include programming languages, operating systems, foundational data structures (such as EDIFACT), generic approaches to organization structuring, foundational structures for organizing IT operations (such as ITIL® or the IT4IT Reference Architecture), etc.

Common Systems Solutions

A Common Systems Solution is an implementation of a Common Systems Architecture comprised of a set of products and services, which may be certified or branded. It represents the highest common denominator for one or more solutions in the industry segments that the Common Systems Solution supports.

Common Systems Solutions represent collections of common requirements and capabilities, rather than those specific to a particular customer or industry. Common Systems Solutions provide organizations with operating environments specific to operational and informational needs, such as high availability transaction processing and scalable data warehousing systems. Examples of Common Systems Solutions include an enterprise management system product or a security system product.

Computer systems vendors are the typical providers of technology-centric Common Systems Solutions. "Software as a service" vendors are typical providers of common application solutions. Business process outsourcing vendors are typical providers of business capability-centric Common Systems Solutions.

Industry Solutions

An Industry Solution is an implementation of an Industry Architecture, which provides re-usable packages of common components and services specific to an industry.

Fundamental components are provided by Common Systems Solutions and/or Foundation Solutions, and are augmented with industry-specific components. Examples include: a physical database schema or an industry-specific point-of-service device.

Industry Solutions are industry-specific, aggregate procurements that are ready to be tailored to an individual organization’s requirements.

In some cases an industry solution may include not only an implementation of the Industry Architecture, but also other solution elements, such as specific products, services, and systems solutions that are appropriate to that industry.

Organization-Specific Solutions

An Organization-Specific Solution is an implementation of the Organization-Specific Architecture that provides the required business capabilities. Because solutions are designed for specific business operations, they contain the highest amount of unique content in order to accommodate the varying people and processes of specific organizations.

Building Organization-Specific Solutions on Industry Solutions, Common Systems Solutions, and Foundation Solutions is the primary purpose of connecting the Architecture Continuum to the Solutions Continuum, as guided by the architects within an enterprise.

An Organization-Specific Solution will be structured in order to support specific SLAs to ensure support of the operational systems at desired service levels. For example, a third-party application hosting provider may offer different levels of support for operational systems. These agreements would define the terms and conditions of that support.

Other key factors to be defined within an Organization-Specific Solution are the key operating parameters and quality metrics that can be used to monitor and manage the environment.

The Enterprise Continuum can provide a key link between architecture, development, and operations personnel by allowing them to communicate and reach agreement on anticipated operational support requirements. Operations personnel can in turn access the Enterprise

Continuum to obtain information regarding the operation concepts and service support requirements of the deployed system.

6.5 The Enterprise Continuum and the ADM

The TOGAF ADM describes the process of developing an enterprise-specific architecture and an enterprise-specific solution(s) which conform to that architecture by adopting and adapting (where appropriate) generic architectures and solutions (left to right in the continuum classification). In a similar fashion, specific architectures and solutions that prove to be credible and effective will be generalized for re-use (right to left in the continuum classification).

At relevant places throughout the TOGAF ADM, there are pointers to useful architecture assets at the relevant level of generality in the continuum classification. These assets can include reference models from The Open Group and industries at large.

The TOGAF Library provides reference models for consideration for use in developing an organization's architecture.

However, in developing architectures in the various domains within an overall Enterprise Architecture, the architect will need to consider the use and re-use of a wide variety of different architecture assets, and the Enterprise Continuum provides an approach for categorizing and communicating these different assets.

6.6 The Enterprise Continuum and Your Organization

The preceding sections have described the Enterprise Continuum, the Architecture Continuum, and the Solutions Continuum. The following sections describe the relationships between each of the three continua and how these relationships should be applied within your organization.

6.6.1 Relationships

Each of the three continua contains information about the evolution of the architectures during their lifecycle:

- The Enterprise Continuum provides an overall context for architectures and solutions, and classifies assets that apply across the entire scope of the enterprise
- The Architecture Continuum provides a classification mechanism for assets that collectively define the architecture at different levels of evolution from generic to specific
- The Solutions Continuum provides the classification for assets to describe specific solutions for the organization that can be implemented to achieve the intent of the architecture

The relationships between the Architecture Continuum and Solutions Continuum are shown in Figure 6-4.

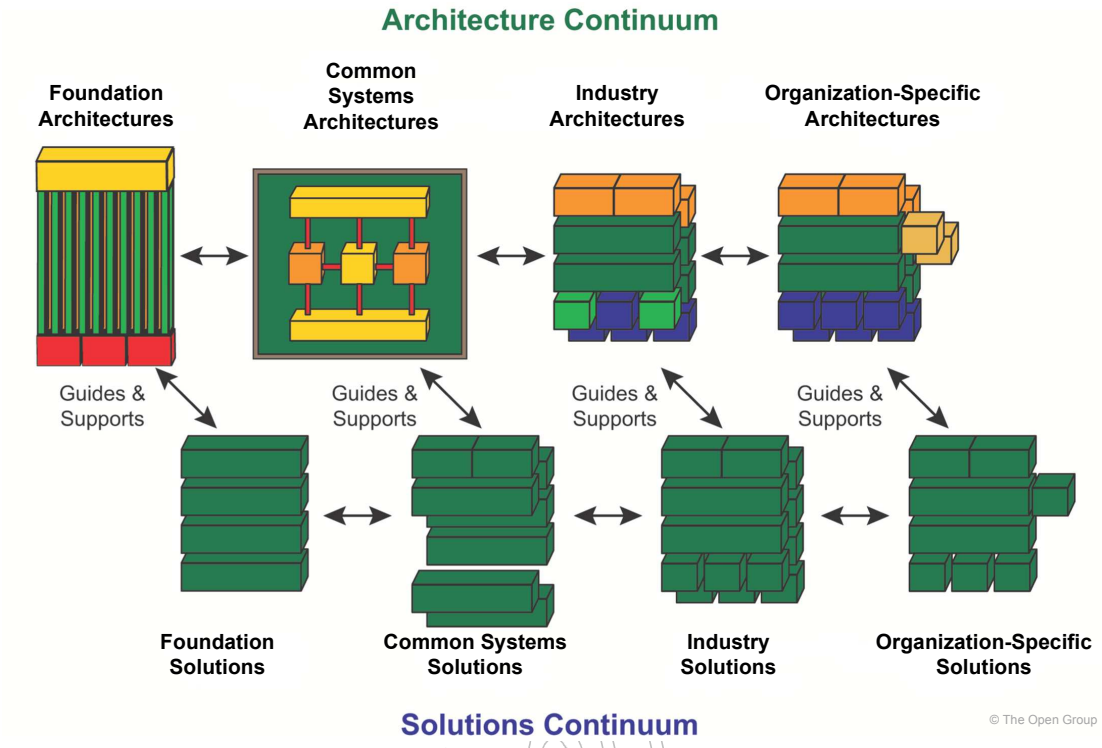

Figure 6-4 Relationships between Architecture and Solutions Continua

The relationship between the Architecture Continuum and the Solutions Continuum is one of guidance, direction, and support. For example, Foundation Architectures guide the creation or selection of Foundation Solutions. Foundation Solutions support the Foundation Architecture by helping to realize the architecture defined in the Architecture Continuum. The Foundation Architecture also guides development of Foundation Solutions, by providing architectural direction, requirements, and principles that guide selection, and realization of appropriate solutions. A similar relationship exists between the other elements of the Enterprise Continuum.

The Enterprise Continuum presents mechanisms to help improve productivity through leverage. The Architecture Continuum offers a consistent way to understand the different architectures and their components. The Solutions Continuum offers a consistent way to understand the different products, systems, services, and solutions required.

The Enterprise Continuum should not be interpreted as representing strictly chained relationships. Organization-Specific Architectures could have components from a Common Systems Architecture, and Organization-Specific Solutions could contain Foundation Solutions. The relationships depicted in Figure 6-1 are an illustration showing opportunities for leveraging architecture and solution components.

6.6.2 Your Enterprise

The TOGAF Standard provides a method for you to "architect" the systems in your enterprise. Your architecture organization will have to deal with each type of architecture described above. For example, it is recommended that you have your own Foundation Architecture that governs all of your systems. You should also have your own Common Systems Architectures that govern major shared systems — such as the networking system or management system. You may have your own industry-specific architectures that govern the way your systems must behave within your industry. Finally, any given department or organization within your business may need its own individual Organization-Specific Architecture to govern the systems within that department.

Your architecture organization will either adopt or adapt existing architectures, or will develop its own architectures from the ground up. In either case, the TOGAF Standard is a tool to help. It provides a method to assist you in generating/maintaining any type of architecture within the Architecture Continuum while leveraging architecture assets already defined, internal or external to your organization. The TOGAF ADM helps you to re-use architecture assets, making your architecture organization more efficient and effective.

Chapter 7: Architecture Repository

7.1 Overview

Operating a mature Architecture Capability within a large enterprise creates a huge volume of architectural output. Effective management and leverage of these architectural work products require a formal taxonomy for different types of architectural asset alongside dedicated processes and tools for architectural content storage.

This section provides a structural framework for an Architecture Repository that allows an enterprise to distinguish between different types of architectural assets that exist at different levels of abstraction in the organization. This Architecture Repository is one part of the wider Enterprise Repository, which provides the capability to link architectural assets to components of the Detailed Design, Deployment, and Service Management Repositories.

At a high level, the following classes of architectural information are expected to be held within an Architecture Repository:

- The **Architecture Metamodel** describes the organizationally tailored application of an architecture framework, including a method for architecture development and a metamodel for architecture content
- The **Architecture Capability** defines the parameters, structures, and processes that support governance of the Architecture Repository
- The **Architecture Landscape** presents an architectural representation of assets in use, or planned, by the enterprise at particular points in time
- The **Standards Library** captures the standards with which new architectures must comply, which may include industry standards, selected products and services from suppliers, or shared services already deployed within the organization
- The **Reference Library** provides guidelines, templates, patterns, and other forms of reference material that can be leveraged in order to accelerate the creation of new architectures for the enterprise
- The **Governance Repository** provides a record of governance activity across the enterprise
- The **Architecture Requirements Repository** provides a view of all authorized architecture requirements which have been agreed with the Architecture Board
- The **Solutions Landscape** presents an architectural representation of the SBBs supporting the Architecture Landscape which have been planned or deployed by the enterprise

The relationships between these areas of the Architecture Repository are shown in Figure 7-1.

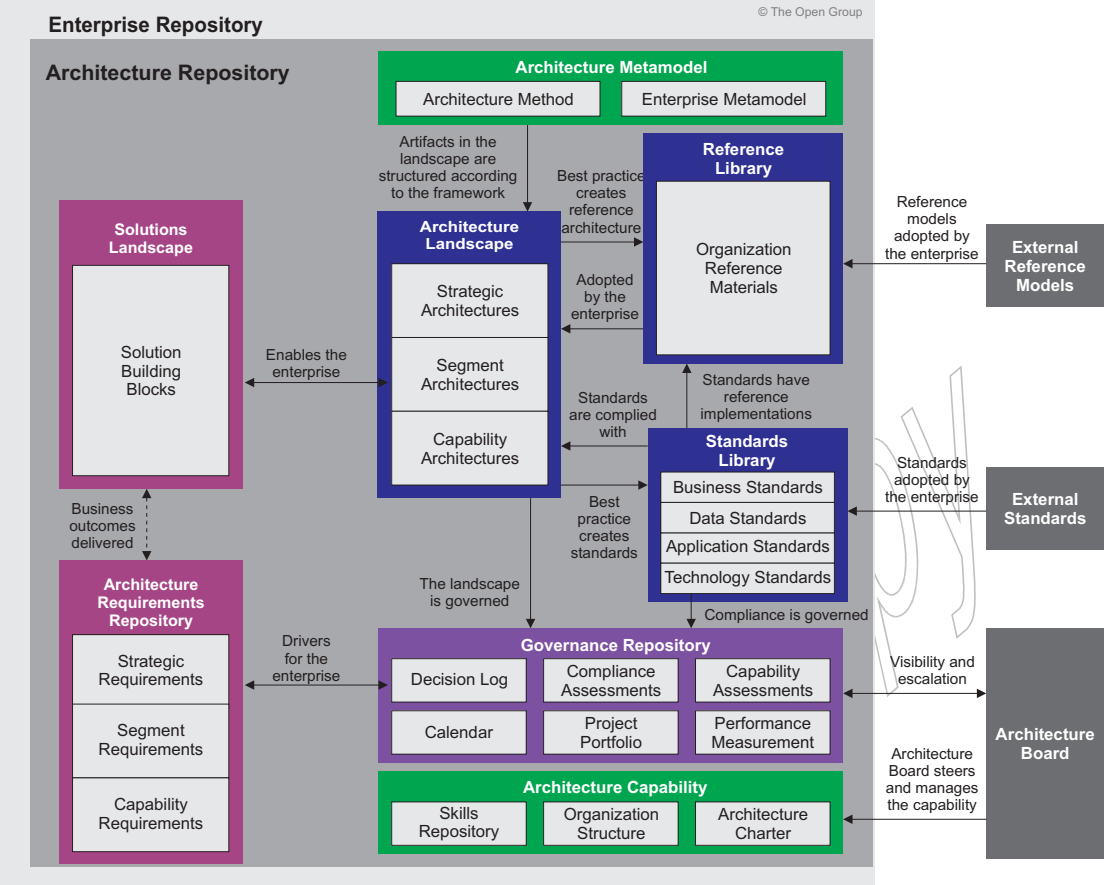

Figure 7-1 Overview of Architecture Repository

The following sections describe the structure and content of the repository areas.

7.2 Architecture Landscape

The Architecture Landscape holds architectural views of the state of the enterprise at particular points in time. Due to the sheer volume and the diverse stakeholder needs throughout an entire enterprise, the Architecture Landscape is divided into three levels of granularity:

1. **Strategic Architectures** (see the TOGAF Standard — Architecture Development Method) show a long-term summary view of the entire enterprise. Strategic Architectures provide an organizing framework for operational and change activity and allow for direction setting at an executive level.
2. **Segment Architectures** (see the TOGAF Standard — Architecture Development Method) provide more detailed operating models for areas within an enterprise. Segment Architectures can be used at the program or portfolio level to organize and operationally align more detailed change activity.

3. **Capability Architectures** (see the TOGAF Standard — Architecture Development Method) show in a more detailed fashion how the enterprise can support a particular unit of capability. Capability Architectures are used to provide an overview of current capability, target capability, and capability increments and allow for individual work packages and projects to be grouped within managed portfolios and programs.

7.3 Reference Library

7.3.1 Overview

The Reference Library provides a repository to hold reference materials that should be used to develop architectures. Reference materials held may be obtained from a variety of sources, including:

- Standards bodies
- Product and service vendors
- Industry communities or forums
- Standard templates
- Enterprise best practice

The Reference Library should contain:

- Reference Architectures
- Reference Models
- Viewpoint Library
- Templates

Note: The terms *reference architecture* and *reference model* are not used carefully in most literature. Reference architecture and reference model have the same relationship as architecture and model. Either can exist as either generic or an organization-specific state. Typically, a generic reference architecture provides the architecture team with an outline of their organization-specific reference architecture that will be customized for a specific organization. For example, a generic reference architecture may identify that there is a need for data models. An example of a reference architecture is the IT4IT Reference Architecture which also defines a common information model for IT management. Another example is the TM Forum eTOM and SID as an organization-specific reference architecture.

In order to segregate different classes of architecture reference materials, the Reference Library can use the Architecture Continuum as a method for classification.

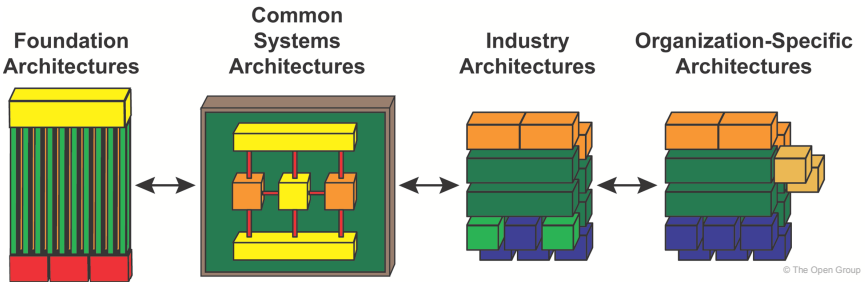

Figure 7-2 Architecture Continuum

The Architecture Continuum, as shown in Figure 7-2, can be viewed as a Reference Library classification scheme. As such it illustrates how reference architectures can be organized across a range — from Foundation Architectures, and Industry-Specific Architectures, to an Organization-Specific Architecture.

The enterprise needs and business requirements are addressed in decreasing abstraction from left to right. The architect will typically find more re-usable architectural elements toward the left of the range. When elements are not found, the requirements for the missing elements are passed to the left of the range for incorporation.

Through this exercise it is important to keep in mind the concepts of levels and partitions. At different levels of granularity there may exist reference materials appropriate to the level, and partitions within the Architecture Landscape can be expected to use different reference material (see the TOGAF Standard — Applying the ADM).

7.4 Standards Library

7.4.1 Overview

The Standards Library provides a repository area to hold a set of specifications, to which architectures must conform. Establishment of a Standards Library provides an unambiguous basis for Architecture Governance because:

- The standards are easily accessible to projects and therefore the obligations of the project can be understood and planned for
- Standards are stated in a clear and unambiguous manner, so that compliance can be objectively assessed

7.4.2 Types of Standard

Standards typically fall into three classes:

- **Legal and Regulatory Obligations**: these standards are mandated by law and therefore an enterprise must comply or face serious consequences
- **Industry Standards**: these standards are established by industry bodies, such as The Open Group, and are then selected by the enterprise for adoption

  Industry Standards offer potential for interoperation and sharing across enterprises, but also fall outside of the control of the enterprise and therefore must be actively monitored.
- **Organizational Standards**: these standards are set within the organization and are based on business aspiration (e.g., selection of standard applications to support portfolio consolidation)

  Organizational Standards require processes to allow for exemptions and standards evolution.

7.4.3 Standards Lifecycle

Standards do not generally exist for all time. New standards are identified and managed through a lifecycle process.

Typically, standards pass through the following stages:

- **Proposed Standard**: a potential standard has been identified for the organization, but has not yet been evaluated for adoption
- **Provisional Standard** (also known as a **Trial Standard**): a Provisional Standard has been identified as a potential standard for the organization, but has not been tried and tested to a level where its value is fully understood

  Projects wishing to adopt Provisional Standards may do so, but under specific pilot conditions, so that the viability of the standard can be examined in more detail.
- **Standard** (also known as an **Active Standard**): a Standard defines a mainstream solution that should generally be used as the approach of choice
- **Phasing-Out Standard** (also known as a **Deprecated Standard**): a Phasing-Out Standard is approaching the end of its useful lifecycle

  Projects that are re-using existing components can generally continue to make use of Phasing-Out Standards. Deployment of new instances of the Phasing-Out Standard is generally discouraged.
- **Retired Standard** (also known as an **Obsolete Standard**): a Retired Standard is no longer accepted as valid within the landscape

  In most cases, remedial action should be taken to remove the Retired Standard from the landscape. Change activity on a Retired Standard should only be accepted as a part of an overall decommissioning plan.

All standards should be periodically reviewed to ensure that they sit within the right stage of the standards lifecycle. As a part of standards lifecycle management, the impact of changing the lifecycle status should be addressed to understand the landscape impact of a standards change and plan for appropriate action to address it.

7.4.4 Standards Classification within the Standards Library

Standards within the Standards Library are categorized according to the building blocks within the TOGAF Enterprise Metamodel. Each metamodel entity can potentially have standards associated with it (e.g., Business Service, Technology Component).

Standards may relate to "approved" building blocks (e.g., a list of standard technology components) or may specify appropriate use of a building block (e.g., scenarios where messaging infrastructure is appropriate, application communication standards are defined).

At the top level, standards are classified in line with the TOGAF architecture domains, including the following areas:

- **Business Standards**:
  - Standard shared business capabilities
  - Standard role and actor definitions
  - Security and governance standards for business activity
- **Data Standards**:
  - Standard coding and values for data
  - Standard structures and formats for data
  - Standards for origin and ownership of data
  - Restrictions on replication and access
- **Applications Standards**:
  - Standard/shared applications supporting specific business functions
  - Standards for application communication and interoperation
  - Standards for access, presentation, and style
- **Technology Standards**;
  - Standard hardware products
  - Standard software products
  - Standards for software development

7.5 Governance Repository

7.5.1 Overview

The Governance Repository provides a repository area to hold shared information relating to the ongoing governance of projects. Maintaining a shared repository of governance information is important, because:

- Decisions made during projects (such as standards deviations or the rationale for a particular architectural approach) are important to retain and access on an ongoing basis

  For example, if a system is to be replaced, having sight of the key architectural decisions that shaped the initial implementation is highly valuable as it will highlight constraints that may otherwise be obscured.

- Many stakeholders are interested in the outcome of project governance (e.g., other projects, customers of the project, the Architecture Board, etc.)

7.5.2 Contents of the Governance Repository

The Governance Repository should contain the following items:

- **Decision Log**: a log of all architecturally significant decisions that have been made in the organization

  This would typically include:

  — Product selections

  — Justification for major architectural features of projects

  — Standards deviations

  — Standards lifecycle changes

  — Change Request evaluations and approvals

  — Re-use assessments

- **Compliance Assessments**: at key checkpoint milestones in the progress of a project, a formal architecture review will be carried out

  This review will measure the compliance of the project to the defined architecture standards. For each project, this log should include:

  — Project overview

  — Progress overview (timeline, status, issues, risks, dependencies, etc.)

  — Completed architecture checklists

  — Standards compliance assessment

  — Recommended actions

- **Capability Assessments**: depending on their objectives, some projects will carry out assessments of business, IT, or Architecture Capability

  These assessments should be periodically carried out and tracked to ensure that appropriate progress is being made. This log should include:

- Templates and reference models for executing Capability Assessments
- Business Capability Assessments
- IT capability, maturity, and impact assessments
- Architecture maturity assessments

- **Calendar**: the Calendar should show a schedule of in-flight projects and formal review sessions to be held against these projects
- **Project Portfolio**: the Project Portfolio should hold summary information about all in-flight projects that fall under Architecture Governance, including:
  - The name and description of the project
  - Architectural scope of the project
  - Architectural roles and responsibilities associated with the project
- **Performance Measurement**: based on a charter for the architecture function, a number of performance criteria will typically be defined

  The Performance Measurement log should capture metrics relating to project governance and any other performance metrics relating to the architecture charter so that performance can be measured and evaluated on an ongoing basis.

7.6 The Architecture Requirements Repository

7.6.1 Overview

The Architecture Requirements Repository is used by all phases of the ADM to record and manage all information relevant to the architecture requirements. The requirements address the many types of architecture requirements; i.e., strategic, segment, and capability requirements which are the major drivers for the Enterprise Architecture.

Requirements can be gathered at every stage of the architecture development cycle and need to be approved through the various phases and governance processes.

The Requirements Management phase is responsible for the management of the contents of the Architecture Requirements Repository and ensuring the integrity of all requirements and their availability for access by all phases.

7.6.2 Contents of the Architecture Requirements Repository

The Architecture Requirements Repository holds architectural requirements of the required state of the enterprise at particular points in time. Due to the sheer volume and the diverse stakeholder needs throughout the architecture development cycle, the Architecture Requirements are divided into three levels of granularity:

1. **Strategic Architecture Requirements** show a long-term summary view of the requirements for the entire enterprise.

   Strategic Architecture Requirements identify operational and change requirements for direction setting at an executive level.

2. **Segment Architecture Requirements** provide more detailed operating model requirements for areas within an enterprise.

   Segment Architecture Requirements may identify requirements at the program or portfolio level to identify and align more detailed change activity.

3. **Capability Architecture Requirements** identify the detailed requirements for a particular unit of capability.

   Capability Architecture Requirements identify requirements for individual work packages and projects to be grouped within managed portfolios and programs.

The business outcomes for architecture requirements will be reflected in the Solutions Landscape over time. When this occurs, the architecture requirements are met and archived for audit purposes.

7.7 Solutions Landscape

The Solutions Landscape holds the SBBs which support the ABBS specified, developed, and deployed. The building blocks may be products or services which may be categorized according to the Enterprise Continuum categorization and/or the ABB specifications as Strategic, Segment, or Capability SBBs.

SBBs may also include tools, systems, services, and information which describe the actual solutions that may be selected and their operation. For example, vendor-specific reference models or vendor-specific Levels 4 and 5 of the IT4IT Reference Architecture would be defined here.

However, the Solutions Landscape will not include the information and data content produced by the solutions selected; that is the responsibility of the solutions themselves.

7.8 The Enterprise Repository

While the Architecture Repository holds information concerning the Enterprise Architecture and associated specifications and artifacts, there are a considerable number of enterprise repositories that support the architecture both inside and outside of the enterprise.

These can include development repositories, specific operating environments, instructions, and configuration management repositories.

7.9 External Repositories

7.9.1 External Reference Models

There are many industry reference models available which may assist in understanding the role of and developing the reference architectures.

7.9.2 External Standards

These relate to industry, best practice, or formal defined standards used by leading organizations. Examples include ISO, IEEE, and Government standards.

7.9.3 Architecture Board Approvals

Decisions made by the Architecture Board which affect the Enterprise Architecture are often recorded in the minutes of meetings. These minutes are often held in documentation archives which are excluded from the Architecture Repository for legal or regulatory reasons.

Index

Index