The Open Group Standard

The TOGAF® Standard, Version 9.2

The Open Group

The Open Group Standard

The TOGAF® Standard, Version 9.2

ISBN: 1-947754-11-9
Document Number: C182

Published in the U.S. by The Open Group, 2005-2018.

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

OGspecs@opengroup.org

Contents

List of Figures

List of Tables

Preface

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 Standard, Version 9.2 is an update to the TOGAF 9.1 standard to provide additional guidance, correct errors, address some structural challenges, and remove obsolete content. All of these changes will make the TOGAF framework easier to use and maintain.[2]

The TOGAF Documentation

The TOGAF documentation consists of a set of documents:

- The TOGAF standard (this document) which describes the generally applicable approach to Enterprise and IT Architecture
- The TOGAF Library, a portfolio of guidance material to support the practical application of the TOGAF approach

This Document

There are six parts to this document:

PART I (Introduction) This part provides a high-level introduction to the key concepts of Enterprise Architecture and in particular the TOGAF approach. It contains the definitions of terms used throughout the TOGAF documentation.

PART II (Architecture Development Method) This part is the core of the TOGAF framework. It describes the TOGAF Architecture Development Method (ADM) — a step-by-step approach to developing an Enterprise Architecture.

PART III (ADM Guidelines & Techniques) This part contains a collection of guidelines and techniques available for use in applying the TOGAF approach and the TOGAF ADM.

PART IV (Architecture Content Framework) This part describes the TOGAF content framework, including a structured metamodel for architectural artifacts, the use of re-usable Architecture Building Blocks (ABBs), and an overview of typical architecture deliverables.

---

1. The TOGAF Library (https://publications.opengroup.org/togaf-library) provides a publicly available structured list of Guides and White Papers which provide guidance in the practical application of the TOGAF approach.

2. A full comparison of this version with the TOGAF Version 9.1 standard may be found in The Open Group White Paper: An Introduction to the TOGAF® Standard, Version 9.2 (www.opengroup.org/library/w182).

PART V (Enterprise Continuum & Tools) This part discusses appropriate taxonomies and tools to categorize and store the outputs of architecture activity within an enterprise.

PART VI (Architecture Capability Framework) This part discusses the organization, processes, skills, roles, and responsibilities required to establish and operate an architecture function within an enterprise.

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.

Keywords

architecture, architecture framework, architecture development method, architect, architecting, enterprise architecture, enterprise architecture framework, enterprise architecture method, method, methods, open, group, technical reference model, standards, standards information base

About The Open Group

The Open Group

The Open Group is a global consortium that enables the achievement of business objectives through technology standards. Our diverse membership of more than 580 organizations includes customers, systems and solutions suppliers, tools vendors, integrators, academics, and consultants across multiple industries.

The Open Group aims to:

- Capture, understand, and address current and emerging requirements, establish policies, and share best practices
- Facilitate interoperability, develop consensus, and evolve and integrate specifications and open source technologies
- Operate the industry's premier certification service

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

The Open Group publishes a wide range of technical documentation, most of which is focused on development of Open Group Standards and Guides, but which also includes white papers, technical studies, certification and testing documentation, and business titles.

Full details and a catalog are available at www.opengroup.org/library.

Trademarks

ArchiMate®, DirecNet®, Making Standards Work®, OpenPegasus®, Platform 3.0®, The Open Group®, TOGAF®, UNIX®, UNIXWARE®, X/Open®, and the Open Brand X® logo are registered trademarks and Boundaryless Information Flow™, Build with Integrity Buy with Confidence™, Dependability Through Assuredness™, EMMM™, FACE™, the FACE™ logo, IT4IT™, the IT4IT™ logo, O-DEF™, O-PAS™, Open FAIR™, Open Platform 3.0™, Open Process Automation™, Open Trusted Technology Provider™, SOSA™, the Open O™ logo, and The Open Group Certification logo (Open O and check™) are trademarks of The Open Group.

CMMI®, IPD-CMM®, P-CMM®, SA-CMM®, SCAMPI®, SE-CMM®, and SW-CMM® are registered trademarks of the Software Engineering Institute (SEI), Carnegie Mellon University.

COBIT® is a registered trademark of the Information Systems Audit and Control Association (ISACA) and the IT Governance Institute.

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

FICO® is a registered trademark of Fair Isaac Corporation in the United States and other countries.

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

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

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

MDA®, Model-Driven Architecture®, Object Management Group®, OMG®, SysML®, and UML® are registered trademarks and BMM™, BPMN™, Business Motivation Model™, Business Process Modeling Notation™, ITPMF™, IT Portfolio Management Facility™, SPEM™, Systems Modeling Language™, and Unified Modeling Language™ are trademarks of the Object Management Group.

Merriam-Webster Collegiate Dictionary® is a registered trademark of Merriam-Webster, Incorporated.

OASIS® is a registered trademark of OASIS, the open standards consortium.

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

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.

Participants

This document was prepared by The Open Group Architecture Forum. When The Open Group approved the TOGAF Standard, Version 9.2 on March 5, 2018, the Architecture Forum officers were as follows:

Mike Lambert, Fellow of The Open Group, Chair
Robert Weisman, Build The Vision, Vice-Chair
Sonia Gonzalez, The Open Group, Architecture and ArchiMate Forum Director

Architecture Forum Technical Reviewers

Technical reviewers are those individuals who have submitted comments during the company review, or participated in an issue resolution meeting during the development of the TOGAF Standard, Version 9.2:

Rob Akershoek, DXC Technology
Emmanuel Amamoo-Otchere, Huawei Technologies, Co. Ltd.
Michael Anniss, M.J. Anniss Ltd.
Graham Berrisford, Avancier
Terence Blevins, Fellow of The Open Group
Bill Estrem, Metaplexity Associates Inc.
Abdallah Fateen, Microsoft
Michael Fulton, Nationwide
Sonia Garnica, Intelligent Training de Colombia
Mats Gejnevall, Innovate
Kirk Hansen, Metaplexity Associates Inc.
Laura Hart, The MITRE Corporation
Judith Jones, ATE Enterprises
Andrew Josey, The Open Group
Rolf Knoll, NovaTec Consulting GmbH
J. Bryan Lail, Raytheon
Mike Lambert, Fellow of The Open Group
Neil Levette, TRM Technologies Inc.
Kenneth Lind, XLENT IT Consulting
Stephen Marshall, IBM
Chalon Mullins, Business Architecture Guild
Alain Picard, Benchmark Consulting
Sriram Sabesan, Conexiam
Satish Shettar, IBM
Robert Weisman, Build The Vision
Paul Williams, Capgemini

Architecture Forum Members

An up-to-date list of Forum members can be found at: www.opengroup.org/architecture.

Acknowledgements

The Open Group gratefully acknowledges The Open Group Architecture Forum for developing the TOGAF standard.

The Open Group gratefully acknowledges the contribution of the US Air Force for its Headquarters Air Force Principles.

The Open Group gratefully acknowledges those past and present members of the Architecture Forum who have served as its officers (Chairs and Vice-Chairs) since its inception. In alphabetical order:

Mick Adams
Christer Askerfjord
Terence Blevins
Bill Estrem
Hugh Fisher
Chris Forde
Chris Greenslade
Ed Harrington
Peter Haviland
Dave Hornford
David Jackson
Mike Lambert
Stuart Macgregor
Ian McCall
Tara Paider
Barry Smith
Walter Stahlecker
Sheena Thompson
Paul van der Merwe
Dave Van Gelder
Jane Varnus
Vish Viswanathan
Robert Weisman
Hal Wilson

Referenced Documents

The following documents are referenced in the TOGAF standard:

- A Guide to the Project Management Body of Knowledge (PMBOK®) Guide), Project Management Institute; refer to www.pmi.org/pmbok-guide-standards
- A Method for Identifying Process Re-Use Opportunities to Enhance the Operating Model, M. de Vries, A. van der Merwe, P. Kotze, A. Gerber, in proceedings of the 2011 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)
- Analysis Patterns — Reusable Object Models, M. Fowler, ISBN: 0-201-89542-0, Addison-Wesley
- ANSI/IEEE Std 1471-2000: Systems and Software Engineering — Recommended Practice for Architectural Description of Software-intensive Systems
- A Pattern Language: Towns, Buildings, Construction, Christopher Alexander, ISBN: 0-19-501919-9, Oxford University Press, 1979
- Business Capabilities, an Open Group Guide, March 2016 (G161), published by The Open Group; refer to www.opengroup.org/library/g161
- Business Motivation Model™ (BMM™), Object Management Group (OMG); refer to www.omg.org/spec/BMM/About-BMM
- Business Transformation Enablement Program (BTEP), Canadian Government; refer to www.tbs-sct.gc.ca/btep-pto/index_e.asp
- Business Process Modeling Notation™ (BPMN™) Specification, Object Management Group (OMG); refer to www.bpmn.org
- Control Objectives for Information and related Technology (COBIT®), Version 4.0, IT Governance Institute, 2005
- Corporate Governance, Ranami Naidoo, ISBN: 1-919-903-0086, Double Storey, 2002
- Design Patterns: Elements of Reusable Object-Oriented Software, Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, ISBN: 0-201-63361-2, Addison-Wesley, October 1994
- Enterprise Architecture as Strategy, Jeanne Ross, Peter Weill, David C. Robertson, ISBN: 1-59139-839-8, Harvard Business School Press, 2006
- Enterprise Architecture Capability Maturity Model (ACMM), Version 1.2, United States Department of Commerce, December 2007
- Enterprise Architecture Maturity Model, Version 1.3, National Association of State CIOs (NASCIO), December 2003
- Enterprise Architecture Planning (EAP): Developing a Blueprint for Data, Applications, and Technology, Steven H. Spewak, Steven C. Hill, ISBN: 0-47-159985-9, John Wiley & Sons, 1993

- Exploring Synergies between TOGAF® and Frameworx, White Paper, May 2011 (W114), published by The Open Group; refer to www.opengroup.org/library/w114
- Headquarters Air Force Principles for Information Management, US Air Force, June 29, 1998
- Integrating Risk and Security within a TOGAF® Enterprise Architecture, an Open Group Guide, January 2016 (G152), published by The Open Group; refer to www.opengroup.org/library/g152
- Integrating the TOGAF® Standard with the BIAN Service Landscape, White Paper, October 2013 (W135), published by The Open Group; refer to www.opengroup.org/library/w135
- ISO/IEC 20000: 2011, Information Technology — Service Management
- ISO/IEC/IEEE 15288: 2015, Systems and Software Engineering — System Life Cycle Processes
- ISO/IEC/IEEE 42010: 2011, Systems and Software Engineering — Architecture Description
- IT Portfolio Management Facility™ (ITPMF™) Specification, Object Management Group (OMG); refer to www.omg.org/spec/ITPMF
- Merriam-Webster Collegiate Dictionary, Merriam-Webster, English Language, 11th Edition, April 2008, ISBN-10: 0877798095, ISBN-13: 978-0877798095; refer to www.merriam-webster.com
- Model-Driven Architecture® (MDA®) Specification, Object Management (OMG); refer to www.omg.org/mda
- OECD Principles of Corporate Governance, Organization for Economic Co-operation and Development, December 2001; refer to www.oecd.org
- Organigraphs: Drawing How Companies Really Work, H. Mintzberg and L. Van der Heyden, Harvard Business Review, October 1999; refer to https://hbr.org/1999/09/organigraphs-drawing-how-companies-really-work
- Pattern-Oriented Software Architecture: A System of Patterns, F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, M. Stal, ISBN: 0-471-95869-7, John Wiley & Sons, 1996
- Patterns and Software: Essential Concepts and Terminology, Brad Appleton, 2000; refer to www.bradapp.com/docs/patterns-intro.html
- Re-usable Asset Specification (RAS), Version 2.2, Object Management Group (OMG), November 2005; refer to www.omg.org/spec/RAS
- Service Component Architecture (SCA) Specification, developed by the Open Service Oriented Architecture (OSOA) collaboration; refer to www.oasis-opencsa.org/sca
- Service Data Objects (SDO) Specification, developed by the Open Service Oriented Architecture (OSOA) collaboration; refer to www.oasis-opencsa.org/sdo
- Software Processing Engineering Metamodel (SPEM™) Specification, Version 2.0, Object Management Group (OMG), April 2008; refer to www.omg.org/spec/SPEM
- Systems Modeling Language™ (SysML®), Object Management Group (OMG); refer to www.sysml.org
- The Art of Systems Architecting, Eberhardt Rechtin, Mark W. Maier, 2000
- The Open Group IT4IT™ Reference Architecture, Version 2.1, an Open Group Standard, January 2017 (C171), published by The Open Group; refer to www.opengroup.org/library/c171
- The Oregon Experiment, Christopher Alexander, ISBN: 0-19-501824-9, Oxford University Press, 1975
- The Timeless Way of Building, Christopher Alexander, ISBN: 0-19-502402-8, Oxford University Press, 1979

- TOGAF® 9 and DoDAF 2.0, White Paper, July 2010 (W105), published by The Open Group; refer to www.opengroup.org/library/w105
- TOGAF® and SABSA® Integration, White Paper, October 2011 (W117), published by The Open Group; refer to www.opengroup.org/library/w117
- TOGAF® Series Guide: Business Scenarios, September 2017 (G176), published by The Open Group; refer to www.opengroup.org/library/g176
- TOGAF® Series Guide: The TOGAF Integrated Information Infrastructure Reference Model (III-RM): An Architected Approach to Boundaryless Information Flow™, November 2017 (G179), published by The Open Group; refer to www.opengroup.org/library/g179
- TOGAF® Series Guide: The TOGAF Technical Reference Model (TRM), September 2017 (G175), published by The Open Group; refer to www.opengroup.org/library/g175
- TOGAF® Series Guide: Using the TOGAF® Framework to Define and Govern Service-Oriented Architectures, September 2017 (G174), published by The Open Group; refer to www.opengroup.org/library/g174
- TOGAF® Series Guide: Value Streams, October 2017 (G178), published by The Open Group; refer to www.opengroup.org/library/g178
- UML Profile and Metamodel for Services (UPMS) RFP (OMG soa/2006-09-09), Object Management Group (OMG), June 2007
- Unified Modeling Language™ (UML®) Specification, Object Management Group (OMG); refer to www.uml.org

The following websites provide useful reference material:

- The Cloud Computing Design Patterns community website (refer to www.cloudpatterns.org)
- The Information Technology Governance Institute: www.isaca.org/About-ISACA/IT-Governance-Institute

  This website has many resources that can help with corporate assessment of both IT and governance in general.
- The Patterns Home Page: hillside.net/patterns

  This website is hosted by The Hillside Group and provides information about patterns, links to online patterns, papers, and books dealing with patterns, and patterns-related mailing lists.
- The Patterns-Discussion FAQ: g.oswego.edu/dl/pd-FAQ/pd-FAQ.html

  This website is maintained by Doug Lea and provides a thorough and highly readable FAQ about patterns.
- The SOA Patterns community website (refer to www.soapatterns.org/), dedicated to the ongoing development and expansion of the SOA design pattern catalog
- The Volere website has a useful list of leading requirements tools www.volere.co.uk/tools.htm

The TOGAF Standard, Version 9.2

Part I:

Introduction

The Open Group

Chapter 1

Introduction

The TOGAF standard is a framework for Enterprise Architecture. It may be used freely by any organization wishing to develop an Enterprise Architecture for use within that organization (see Section 1.4.1).

The TOGAF standard is developed and maintained by members of The Open Group, working within the Architecture Forum (refer to www.opengroup.org/architecture). The original development of TOGAF Version 1 in 1995 was based on the Technical Architecture Framework for Information Management (TAFIM), developed by the US Department of Defense (DoD). The DoD gave The Open Group explicit permission and encouragement to create Version 1 of the TOGAF standard by building on the TAFIM, which itself was the result of many years of development effort and many millions of dollars of US Government investment.

Starting from this sound foundation, the members of The Open Group Architecture Forum have developed successive versions of the TOGAF standard and published each one on The Open Group public website.

This version builds on previous versions of the TOGAF standard and updates the material available to architecture practitioners to assist them in building a sustainable Enterprise Architecture. Work on White Papers and Guides describing how to to integrate and use this standard with other frameworks and architectural styles has highlighted the universal framework parts of the standard, as well as industry, architecture style, and purpose-specific tools, techniques, and guidance. This work is embodied in the TOGAF Library.[1]

Although all of the TOGAF documentation works together as a whole, it is expected that organizations will customize it during adoption, and deliberately choose some elements, customize some, exclude some, and create others. For example, an organization may wish to adopt the TOGAF metamodel, but elect not to use any of the guidance on how to develop an in-house Technology Architecture because they are heavy consumers of cloud and Open Platform 3.0™.

Regardless of your prior experience, you are recommended to read the Executive Overview (see Section 1.3), where you will find an outline of The Open Group understanding of Enterprise Architecture and answers to fundamental questions, such as:

- Why is an Enterprise Architecture needed?
- Why use the TOGAF standard as a framework for Enterprise Architecture?

1. The TOGAF Library provides an online publicly available structured list of Guides, White Papers, and other resources. Refer to The Open Group Library at https://publications.opengroup.org/togaf-library.

1.1 Structure of this Document

The structure of this document reflects the structure and content of an Architecture Capability within an enterprise, as shown in Figure 1-1.

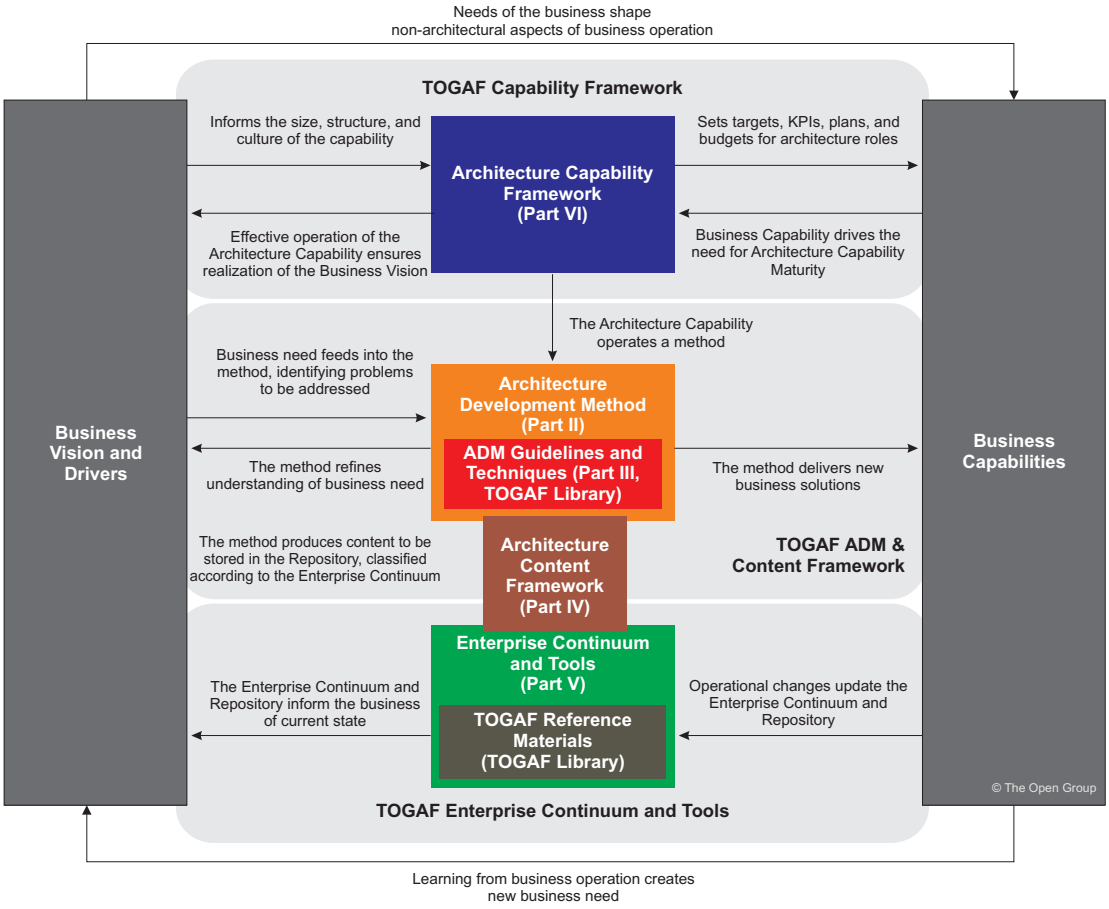

Figure 1-1 Structure of the TOGAF Standard

There are six parts to this document:

PART I (Introduction) This part provides a high-level introduction to the key concepts of Enterprise Architecture and in particular the TOGAF approach. It contains the definitions of terms used throughout this standard.

PART II (Architecture Development Method) This part is the core of the TOGAF framework. It describes the TOGAF Architecture Development Method (ADM) — a step-by-step approach to developing an Enterprise Architecture.

PART III (ADM Guidelines & Techniques) This part contains a collection of guidelines and techniques available for use in applying the TOGAF approach and the TOGAF ADM. Additional guidelines and techniques are available in the TOGAF Library.

PART IV (Architecture Content Framework) This part describes the TOGAF content framework, including a structured metamodel for architectural artifacts, the use of re-usable Architecture Building Blocks (ABBs), and an overview of typical architecture deliverables.

PART V (Enterprise Continuum & Tools) This part discusses appropriate taxonomies and tools to categorize and store the outputs of architecture activity within an enterprise.

PART VI (Architecture Capability Framework) This part discusses the organization, processes, skills, roles, and responsibilities required to establish and operate an architecture function within an enterprise.

The intention of dividing the TOGAF standard into these independent parts is to allow for different areas of specialization to be considered in detail and potentially addressed in isolation. Although all parts work together as a whole, it is also feasible to select particular parts for adoption while excluding others. For example, an organization may wish to adopt the ADM process, but elect not to use any of the materials relating to Architecture Capability.

As an open framework, such use is encouraged, particularly in the following situations:

- Organizations that are new to the TOGAF approach and wish to incrementally adopt TOGAF concepts are expected to focus on particular parts of the specification for initial adoption, with other areas tabled for later consideration
- Organizations that have already deployed architecture frameworks may choose to merge these frameworks with aspects of the TOGAF standard

1.2 Structure of the TOGAF Library

Accompanying this standard is a portfolio of guidance material, known as the TOGAF Library, to support the practical application of the TOGAF approach. The TOGAF Library is a reference library containing guidelines, templates, patterns, and other forms of reference material to accelerate the creation of new architectures for the enterprise.

The TOGAF Library is maintained under the governance of The Open Group Architecture Forum.

Library resources are organized into four sections:

- Section 1. Foundation Documents
- Section 2. Generic Guidance and Techniques
- Section 3. Industry-Specific Guidance and Techniques
- Section 4. Organization-Specific Guidance and Techniques

Where resources within the Library apply to the deployment of the TOGAF ADM and make explicit reference to "anchor points" within the TOGAF standard they are classified within the Library as Dependent documents. Resources that provide guidance on how to utilize features described in the standard are classified as Supporting documents. Resources that relate to Enterprise Architecture in general, and that do not make any specific references to the TOGAF standard, are classified as EA General documents.

1.3 Executive Overview

This section provides an executive overview of Enterprise Architecture, the basic concepts of what it is (not just another name for IT Architecture), and why it is needed. It provides a summary of the benefits of establishing an Enterprise Architecture and adopting the TOGAF approach to achieve that.

What is an enterprise?

The TOGAF standard considers an "enterprise" to be any collection of organizations that have common goals.

For example, an enterprise could be:

- A whole corporation or a division of a corporation
- A government agency or a single government department
- A chain of geographically distant organizations linked together by common ownership
- Groups of countries or governments working together to create common or shareable deliverables or infrastructures
- Partnerships and alliances of businesses working together, such as a consortium or supply chain

The term "Enterprise" in the context of "Enterprise Architecture" can be applied to either an entire enterprise, encompassing all of its business activities and capabilities, information, and technology that make up the entire infrastructure and governance of the enterprise, or to one or more specific areas of interest within the enterprise. In both cases, the architecture crosses multiple systems, and multiple functional groups within the enterprise.

Confusion often arises from the evolving nature of the term "enterprise". An extended enterprise nowadays frequently includes partners, suppliers, and customers. If the goal is to integrate an extended enterprise, then the enterprise comprises the partners, suppliers, and customers, as well as internal business units.

The enterprise operating model concept is useful to determine the nature and scope of the Enterprise Architecture within an organization. Many organizations may comprise multiple enterprises, and may develop and maintain a number of independent Enterprise Architectures to address each one. These enterprises often have much in common with each other including processes, functions, and their information systems, and there is often great potential for wider gain in the use of a common architecture framework. For example, a common framework can provide a basis for the development of common building blocks and solutions, and a shareable Architecture Repository for the integration and re-use of business models, designs, information, and data.

Why is an Enterprise Architecture needed?

The purpose of Enterprise Architecture is to optimize across the enterprise the often fragmented legacy of processes (both manual and automated) into an integrated environment that is responsive to change and supportive of the delivery of the business strategy.

Today's CEOs know that the effective management and exploitation of information and Digital Transformation are key factors to business success, and indispensable means to achieving competitive advantage. An Enterprise Architecture addresses this need, by providing a strategic context for the evolution and reach of digital capability in response to the constantly changing needs of the business environment.

For example, the rapid development of social media, Internet of Things, and cloud computing has radically extended the capacity of the enterprise to create new market opportunities.

Furthermore, a good Enterprise Architecture enables you to achieve the right balance between business transformation and continuous operational efficiency. It allows individual business units to innovate safely in their pursuit of evolving business goals and competitive advantage. At the same time, the Enterprise Architecture enables the needs of the organization to be met with an integrated strategy which permits the closest possible synergies across the enterprise and beyond.

What are the benefits of an Enterprise Architecture?

An effective Enterprise Architecture can bring important benefits to the organization. Specific benefits of an Enterprise Architecture include:

- More effective and efficient business operations:
  - Lower business operation costs
  - More agile organization
  - Business capabilities shared across the organization
  - Lower change management costs
  - More flexible workforce
  - Improved business productivity
- More effective and efficient Digital Transformation and IT operations:
  - Extending effective reach of the enterprise through digital capability
  - Bringing all components of the enterprise into a harmonized environment
  - Lower software development, support, and maintenance costs
  - Increased portability of applications
  - Improved interoperability and easier system and network management
  - Improved ability to address critical enterprise-wide issues like security
  - Easier upgrade and exchange of system components
- Better return on existing investment, reduced risk for future investment:
  - Reduced complexity in the business and IT
  - Maximum return on investment in existing business and IT infrastructure
  - The flexibility to make, buy, or out-source business and IT solutions
  - Reduced risk overall in new investments and their cost of ownership
- Faster, simpler, and cheaper procurement:
  - Buying decisions are simpler, because the information governing procurement is readily available in a coherent plan
  - The procurement process is faster — maximizing procurement speed and flexibility without sacrificing architectural coherence

— The ability to procure heterogeneous, multi-vendor open systems

— The ability to secure more economic capabilities

What specifically would prompt the development of an Enterprise Architecture?

Typically, preparation for business transformation needs or for radical infrastructure changes initiates an Enterprise Architecture review or development. Often key people identify areas of change required in order for new business goals to be met. Such people are commonly referred to as the "stakeholders" in the change. The role of the architect is to address their concerns by:

- Identifying and refining the requirements that the stakeholders have
- Developing views of the architecture that show how the concerns and requirements are going to be addressed
- Showing the trade-offs that are going to be made in reconciling the potentially conflicting concerns of different stakeholders

Without the Enterprise Architecture, it is highly unlikely that all the concerns and requirements will be considered and met.

What is an architecture framework?

An architecture framework is a foundational structure, or set of structures, which can be used for developing a broad range of different architectures. It should describe a method for designing a target state of the enterprise in terms of a set of building blocks, and for showing how the building blocks fit together. It should contain a set of tools and provide a common vocabulary. It should also include a list of recommended standards and compliant products that can be used to implement the building blocks.

Why use the TOGAF standard as a framework for Enterprise Architecture?

The TOGAF standard has been developed through the collaborative efforts of the whole community. Using the TOGAF standard results in Enterprise Architecture that is consistent, reflects the needs of stakeholders, employs best practice, and gives due consideration both to current requirements and the perceived future needs of the business.

Developing and sustaining an Enterprise Architecture is a technically complex process which involves many stakeholders and decision processes in the organization. The TOGAF standard plays an important role in standardizing and de-risks the architecture development process. The TOGAF standard provides a best practice framework for adding value, and enables the organization to build workable and economic solutions which address their business issues and needs.

Who would benefit from using the TOGAF standard?

Any organization undertaking, or planning to undertake, the development and implementation of an Enterprise Architecture for the support of business transformation will benefit from use of the TOGAF standard.

Organizations seeking Boundaryless Information Flow™ can use the TOGAF standard to define and implement the structures and processes to enable access to integrated information within and between enterprises.

Organizations that design and implement Enterprise Architectures using the TOGAF standard are assured of a design and a procurement specification that can facilitate an open systems implementation, thus enabling the benefits of open systems with reduced risk.

1.4 Information on Using the TOGAF Standard

1.4.1 Conditions of Use

The TOGAF standard is freely available for viewing online without a license. Alternatively, it can be downloaded and stored under license, as explained on the TOGAF information website.

In either case, the TOGAF standard can be used freely by any organization wishing to do so to develop an architecture for use within that organization. No part of it may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, for any other purpose including, but not by way of limitation, any use for commercial gain, without the prior permission of the copyright owners.

1.4.2 How Much Does the TOGAF Standard Cost?

The Open Group is committed to delivering greater business efficiency by bringing together buyers and suppliers of information systems to lower the barriers of integrating new technology across the enterprise. Its goal is to realize the vision of Boundaryless Information Flow.

The TOGAF standard is a key part of its strategy for achieving this goal, and The Open Group wants it to be taken up and used in practical architecture projects, and the experience from its use fed back to help improve it.

The Open Group therefore publishes it on its public web server, and allows and encourages its reproduction and use free-of-charge by any organization wishing to use it internally to develop an Enterprise Architecture. (There are restrictions on its commercial use, however; see Section 1.4.1.)

1.4.3 Downloads

Downloads of the TOGAF standard, including printable PDF files, are available under license from the TOGAF information website (refer to www.opengroup.org/togaf/downloads). The license is free to any organization wishing to use the standard entirely for internal purposes (for example, to develop an Enterprise Architecture for use within that organization).

1.5 Why Join The Open Group?

Organizations wishing to reduce the time, cost, and risk of implementing multi-vendor solutions that integrate within and between enterprises need The Open Group as their key partner.

The Open Group brings together the buyers and suppliers of information systems worldwide, and enables them to work together, both to ensure that IT solutions meet the needs of customers, and to make it easier to integrate IT across the enterprise. The TOGAF standard is a key enabler in this task.

Yes, the TOGAF standard itself is freely available. But how much will you spend on developing or updating your Enterprise Architecture? And how much will you spend on procurements based on that architecture? The price of membership of The Open Group is insignificant in comparison with these amounts.

In addition to the general benefits of membership, as a member of The Open Group you will be eligible to participate in The Open Group Architecture Forum, which is the development program within which the TOGAF standard is evolved, and in which TOGAF users come together to exchange information and feedback.

Members of the Architecture Forum gain:

- Immediate access to the fruits of the current TOGAF work program (not publicly available until publication of the next edition of the TOGAF standard) — in effect, the latest information on the standard
- Exchange of experience with other customer and vendor organizations involved in Enterprise Architecture in general, and networking with architects using the TOGAF standard in significant architecture development projects around the world
- Peer review of specific architecture case study material

Chapter 2

Core Concepts

For the purposes of the TOGAF standard, the core concepts provided in this chapter apply.

2.1 What is the TOGAF Standard?

The TOGAF standard is an architecture framework. It provides the methods and tools for assisting in the acceptance, production, use, and maintenance of an Enterprise Architecture. It is based on an iterative process model supported by best practices and a re-usable set of existing architecture assets.

2.2 What is Architecture in the Context of the TOGAF Standard?

ISO/IEC/IEEE 42010: 2011 defines "architecture" as:

> "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."

The TOGAF standard embraces but does not strictly adhere to ISO/IEC/IEEE 42010: 2011 terminology. In addition to the ISO/IEC/IEEE 42010: 2011 definition of "architecture", the TOGAF standard defines a second meaning depending upon the context:

> "The structure of components, their inter-relationships, and the principles and guidelines governing their design and evolution over time."

The TOGAF standard considers the enterprise as a system and endeavors to strike a balance between promoting the concepts and terminology drawn from relevant standards, and commonly accepted terminology that is familiar to the majority of the TOGAF readership. For more on terminology, refer to Chapter 3 and Part IV, Chapter 31.

2.3 What Kind of Architecture Does the TOGAF Standard Deal With?

There are four architecture domains that are commonly accepted as subsets of an overall Enterprise Architecture, all of which the TOGAF standard is designed to support:

- The **Business Architecture** defines the business strategy, governance, organization, and key business processes
- The **Data Architecture** describes the structure of an organization's logical and physical data assets and data management resources

- The **Application Architecture** provides a blueprint for the individual applications to be deployed, their interactions, and their relationships to the core business processes of the organization
- The **Technology Architecture** describes the logical software and hardware capabilities that are required to support the deployment of business, data, and application services; this includes IT infrastructure, middleware, networks, communications, processing, standards, etc.

2.4 Architecture Development Method

The TOGAF Architecture Development Method (ADM) provides a tested and repeatable process for developing architectures. The ADM includes establishing an architecture framework, developing architecture content, transitioning, and governing the realization of architectures.

All of these activities are carried out within an iterative cycle of continuous architecture definition and realization that allows organizations to transform their enterprises in a controlled manner in response to business goals and opportunities.

Phases within the ADM are as follows:

- The **Preliminary Phase** describes the preparation and initiation activities required to create an Architecture Capability including customization of the TOGAF framework and definition of Architecture Principles
- **Phase A: Architecture Vision** describes the initial phase of an architecture development cycle

  It includes information about defining the scope of the architecture development initiative, identifying the stakeholders, creating the Architecture Vision, and obtaining approval to proceed with the architecture development.
- **Phase B: Business Architecture** describes the development of a Business Architecture to support the agreed Architecture Vision
- **Phase C: Information Systems Architectures** describes the development of Information Systems Architectures to support the agreed Architecture Vision
- **Phase D: Technology Architecture** describes the development of the Technology Architecture to support the agreed Architecture Vision
- **Phase E: Opportunities & Solutions** conducts initial implementation planning and the identification of delivery vehicles for the architecture defined in the previous phases
- **Phase F: Migration Planning** addresses how to move from the Baseline to the Target Architectures by finalizing a detailed Implementation and Migration Plan
- **Phase G: Implementation Governance** provides an architectural oversight of the implementation
- **Phase H: Architecture Change Management** establishes procedures for managing change to the new architecture
- **Requirements Management** examines the process of managing architecture requirements throughout the ADM

2.5 Deliverables, Artifacts, and Building Blocks

Architects executing the ADM will produce a number of outputs as a result of their efforts, such as process flows, architectural requirements, project plans, project compliance assessments, etc. The TOGAF Architecture Content Framework (see Part IV, Chapter 29) provides a structural model for architectural content that allows major work products to be consistently defined, structured, and presented.

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, agreed, 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, business interaction matrix, and a use-case 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 required capability and shape the specification of Solution Building Blocks (SBBs); for example, a customer services capability may be required within an enterprise, supported by many SBBs, such as processes, data, and application software

  — **Solution Building Blocks (SBBs)** represent components that will be used to implement the required capability; for example, a network is a building block that can be described through complementary artifacts and then put to use to realize solutions for the enterprise

The relationships between deliverables, artifacts, and building blocks are shown in Figure 2-1.

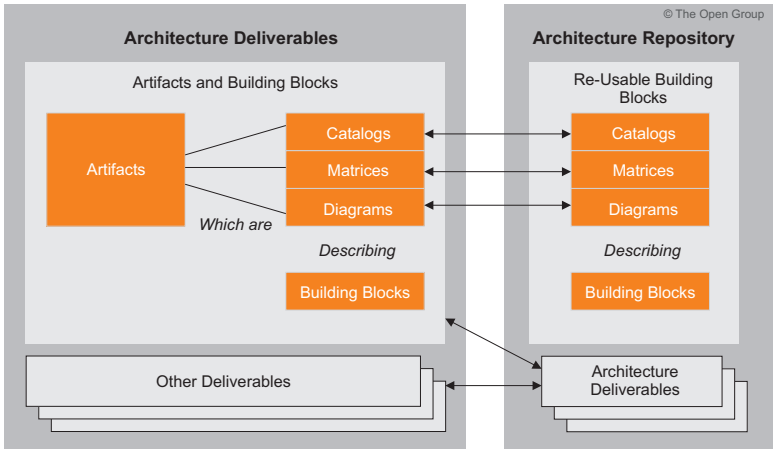

Figure 2-1 Relationships between Deliverables, Artifacts, and Building Blocks

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 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 29-2.

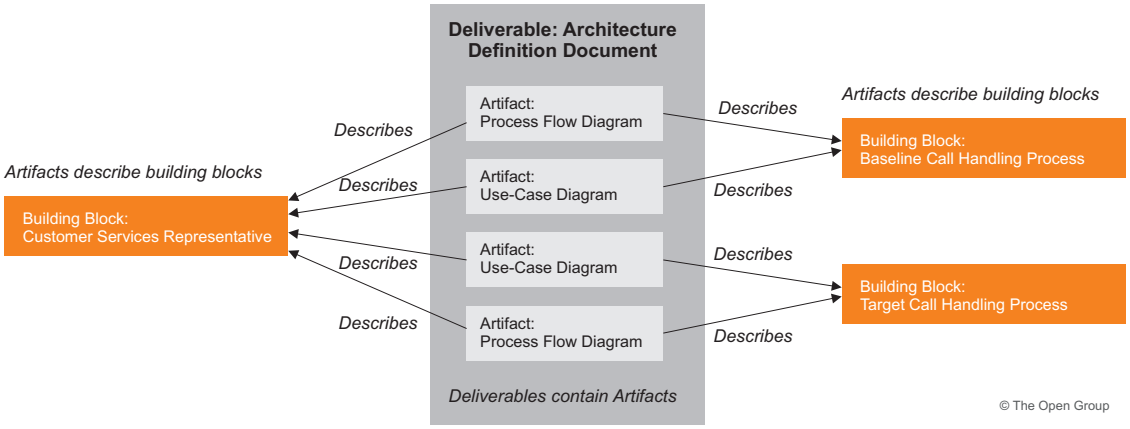

Figure 2-2 Example — Architecture Definition Document

2.6 Enterprise Continuum

The TOGAF standard includes the concept of the Enterprise Continuum, which sets the broader context for an architect and explains how generic solutions can be leveraged and specialized in order to support the requirements of an individual organization. The Enterprise Continuum is a view of the Architecture Repository that provides methods for classifying architecture and solution artifacts as they evolve from generic Foundation Architectures to Organization-Specific Architectures. The Enterprise Continuum comprises two complementary concepts: the Architecture Continuum and the Solutions Continuum.

An overview of the structure and context for the Enterprise Continuum is shown in Figure 2-3.

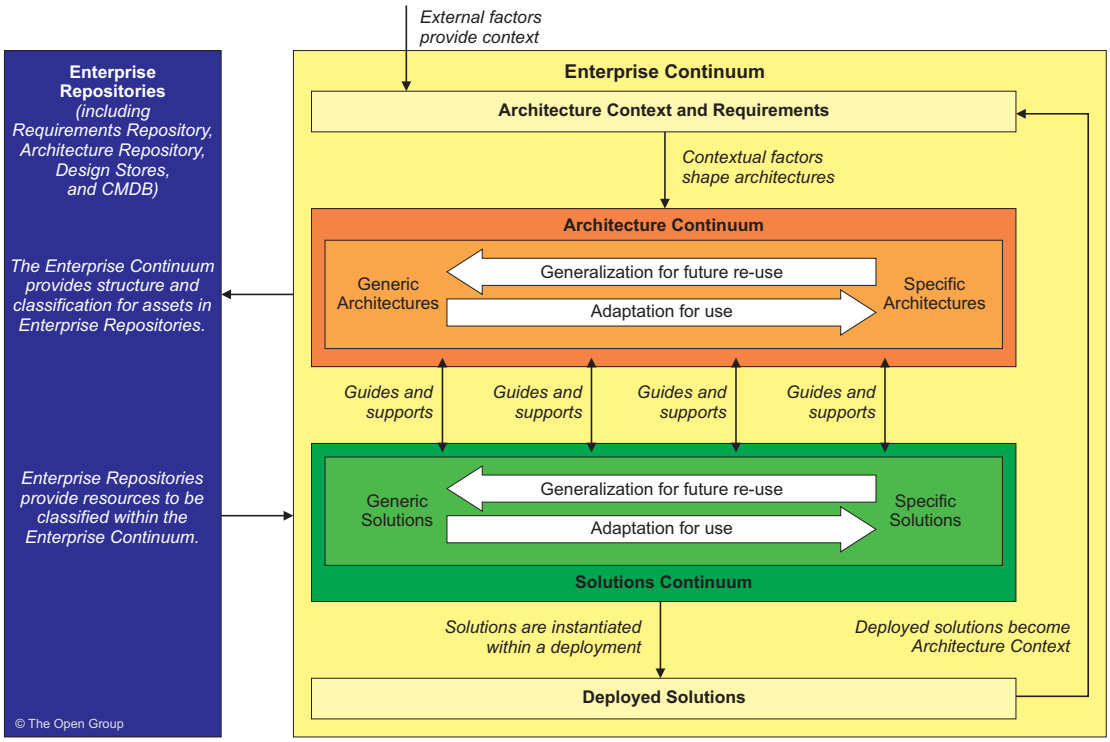

Figure 2-3 Enterprise Continuum

2.7 Architecture Repository

Supporting the Enterprise Continuum is the concept of an Architecture Repository which can be used to store different classes of architectural output at different levels of abstraction, created by the ADM. In this way, the TOGAF standard facilitates understanding and co-operation between stakeholders and practitioners at different levels.

By means of the Enterprise Continuum and Architecture Repository, architects are encouraged to leverage all other relevant architectural resources and assets in developing an Organization-Specific Architecture.

In this context, the TOGAF ADM can be regarded as describing a process lifecycle that operates at multiple levels within the organization, operating within a holistic governance framework and producing aligned outputs that reside in an Architecture Repository. The Enterprise Continuum provides a valuable context for understanding architectural models: it shows building blocks and their relationships to each other, and the constraints and requirements on a cycle of architecture development.

The structure of the TOGAF Architecture Repository is shown in Figure 2-4.

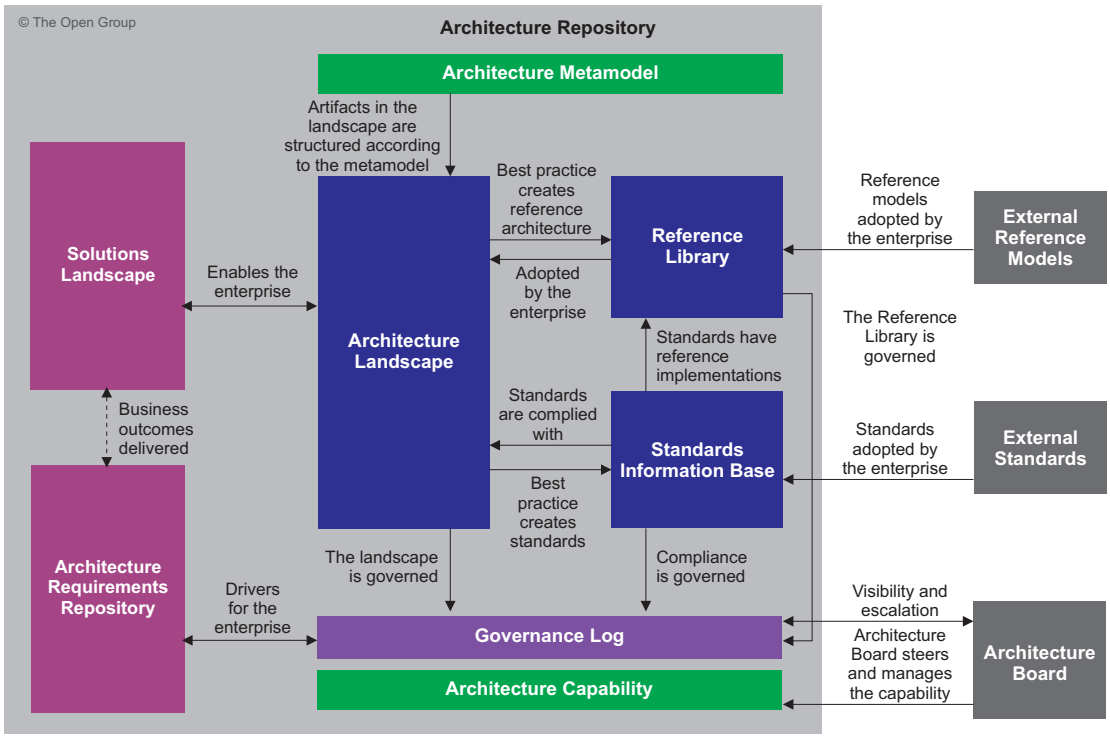

Figure 2-4 TOGAF Architecture Repository Structure

The major components within an Architecture Repository are as follows:

- The **Architecture Metamodel** describes the organizationally tailored application of an architecture framework, including a metamodel for architecture content
- The **Architecture Capability** defines the parameters, structures, and processes that support governance of the Architecture Repository
- The **Architecture Landscape** is the architectural representation of assets deployed within the operating enterprise at a particular point in time — the landscape is likely to exist at multiple levels of abstraction to suit different architecture objectives
- The **Standards Information Base** (SIB) 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 Log** 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

2.8 Establishing and Maintaining an Enterprise Architecture Capability

In order to carry out architectural activity effectively within an enterprise, it is necessary to put in place an appropriate business capability for architecture, through organization structures, roles, responsibilities, skills, and processes. An overview of the TOGAF Architecture Capability is shown in Figure 2-5.

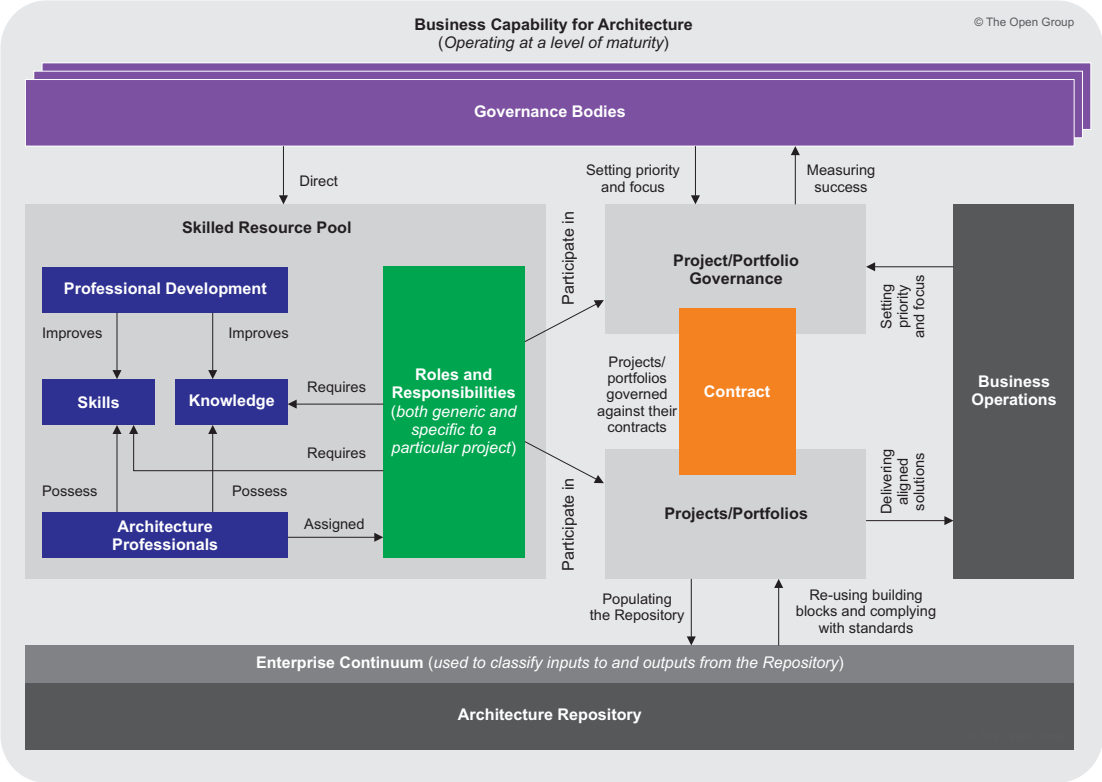

Figure 2-5 TOGAF Architecture Capability Overview

2.9 Establishing the Architecture Capability as an Operational Entity

Barring Architecture Capabilities set up to purely support change delivery programs, it is increasingly recognized that a successful Enterprise Architecture practice must sit on a firm operational footing. In effect, an Enterprise Architecture practice must be run like any other operational unit within a business; i.e., it should be treated like a business. To this end, and over and above the core processes defined within the ADM, an Enterprise Architecture practice should establish capabilities in the following areas:

- Financial Management
- Performance Management
- Service Management
- Risk Management (see Section A.54)
- Resource Management
- Communications and Stakeholder Management (see Section 3.33)
- Quality Management
- Supplier Management (see Section A.60)
- Configuration Management (see Section A.7)
- Environment Management

Central to the notion of operating an ongoing architecture is the execution of well-defined and effective governance, whereby all architecturally significant activity is controlled and aligned within a single framework.

As governance has become an increasingly visible requirement for organizational management, the inclusion of governance within the TOGAF standard aligns the framework with current business best practice and also ensures a level of visibility, guidance, and control that will support all architecture stakeholder requirements and obligations.

The benefits of Architecture Governance include:

- Increased transparency of accountability, and informed delegation of authority
- Controlled risk management
- Protection of the existing asset base through maximizing re-use of existing architectural components
- Proactive control, monitoring, and management mechanisms
- Process, concept, and component re-use across all organizational business units
- Value creation through monitoring, measuring, evaluation, and feedback
- Increased visibility supporting internal processes and external parties' requirements; in particular, increased visibility of decision-making at lower levels ensures oversight at an appropriate level within the enterprise of decisions that may have far-reaching strategic consequences for the organization
- Greater shareholder value; in particular, Enterprise Architecture increasingly represents the core intellectual property of the enterprise — studies have demonstrated a correlation between increased shareholder value and well-governed enterprises

- Integrates with existing processes and methodologies and complements functionality by adding control capabilities

Further detail on establishing an Enterprise Architecture Capability is given in Part VI, Chapter 39.

2.10 Using the TOGAF Standard with Other Frameworks

Two of the key elements of any Enterprise Architecture framework are:

- A definition of the deliverables that the architecting activity should produce
- A description of the method by which this should be done

With some exceptions, the majority of Enterprise Architecture frameworks focus on the first of these — the specific set of deliverables — and are relatively silent about the methods to be used to generate them (intentionally so, in some cases).

Because the TOGAF standard is a generic framework and intended to be used in a wide variety of environments, it provides a flexible and extensible content framework that underpins a set of generic architecture deliverables.

As a result, the TOGAF framework may be used either in its own right, with the generic deliverables that it describes; or else these deliverables may be replaced or extended by a more specific set, defined in any other framework that the architect considers relevant.

In all cases, it is expected that the architect will adapt and build on the TOGAF framework in order to define a tailored method that is integrated into the processes and organization structures of the enterprise. This architecture tailoring may include adopting elements from other architecture frameworks, or integrating TOGAF methods with other standard frameworks or best practices, such as ITIL®, CMMI®, COBIT®, PRINCE2®, PMBOK®, and MSP®. It may also include adopting reference materials from the TOGAF Library, such as the IT4IT™ Reference Architecture. Guidelines for adapting the TOGAF ADM in such a way are given in Part II, Section 4.3.

As a generic framework and method for Enterprise Architecture, the TOGAF standard provides the capability and the collaborative environment to integrate with other frameworks. Organizations are able to fully utilize vertical business domains, horizontal technology areas (such as security or manageability), or application areas (such as e-Commerce) to produce a competitive Enterprise Architecture framework which maximizes their business opportunities.

Chapter 3

Definitions

For the purposes of the TOGAF standard, the following terms and definitions apply. Appendix A should be referenced for supplementary definitions not defined in this chapter. The Merriam-Webster® Collegiate Dictionary should be referenced for terms not defined in this section or Appendix A.

3.1 Abstraction

The technique of providing summarized or generalized descriptions of detailed and complex content.

Note: Abstraction, as in "level of abstraction", can also mean providing a focus for analysis that is concerned with a consistent and common level of detail or abstraction. Abstraction in this sense is typically used in architecture to allow a consistent level of definition and understanding to be achieved in each area of the architecture in order to support effective communication and decision-making. It is especially useful when dealing with large and complex architectures as it allows relevant issues to be identified before further detail is attempted.

3.2 Actor

A person, organization, or system that has one or more roles 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.

Note: In the automotive industry, an original equipment manufacturer would be considered an actor by an automotive dealership that interacts with its supply chain activities.

3.3 Application Architecture

A description of the structure and interaction of the applications as groups of capabilities that provide key business functions and manage the data assets.

Note: Application Architecture is described in Part II, Chapter 10.

3.4 Application Component

An encapsulation of application functionality aligned to implementation structure, which is modular and replaceable. It encapsulates its behavior and data, provides services, and makes them available through interfaces.

Note: For example, a business application such as an accounting, payroll, or CRM system.

An application component usually maintains a data component. It is enabled by technology services provided by technology components.

3.5 Application Platform

The collection of technology components of hardware and software that provide the services used to support applications.

3.6 Architectural Style

The combination of distinctive features related to the specific context within which architecture is performed or expressed; a collection of principles and characteristics that steer or constrain how an architecture is formed.

3.7 Architecture

1. 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. (Source: ISO/IEC/IEEE 42010: 2011)
2. The structure of components, their inter-relationships, and the principles and guidelines governing their design and evolution over time.

3.8 Architecture Building Block (ABB)

A constituent of the architecture model that describes a single aspect of the overall model.

See also Section 3.23.

3.9 Architecture Continuum

A part of the Enterprise Continuum. A repository of architectural elements with increasing detail and specialization.

Note: This Continuum begins with foundational definitions like reference models, core strategies, and basic building blocks. From there it spans to Industry Architectures and all the way to an Organization-Specific Architecture.

See also Section 3.39.

3.10 Architecture Development Method (ADM)

The core of the TOGAF framework. A multi-phase, iterative approach to develop and use an Enterprise Architecture to shape and govern business transformation and implementation projects.

Note: The ADM is described in Part II: Architecture Development Method (ADM).

3.11 Architecture Domain

The architectural area being considered. The TOGAF framework has four primary architecture domains: business, data, application, and technology. Other domains may also be considered (e.g., security).

3.12 Architecture Framework

A conceptual structure used to plan, develop, implement, govern, and sustain an architecture.

3.13 Architecture Governance

The practice of monitoring and directing architecture-related work. The goal is to deliver desired outcomes and adhere to relevant principles, standards, and roadmaps.

See also Section 3.43.

3.14 Architecture Landscape

The architectural representation of assets in use, or planned, by the enterprise at particular points in time.

3.15 Architecture Model

A representation of a subject of interest.

Note: An architecture model provides a smaller scale, simplified, and/or abstract representation of the subject matter.

See also Section 3.72, Section 3.17, and Section 3.18.

3.16 Architecture Principle

A qualitative statement of intent that should be met by the architecture.

Note: A sample set of Architecture Principles is defined in Part III, Chapter 20.

3.17 Architecture View

A representation of a system from the perspective of a related set of concerns.

Note: In some sections of this standard, the term "view" is used as a synonym for "architecture view".

See also Section 3.72 and Section 3.18.

3.18 Architecture Viewpoint

A specification of the conventions for a particular kind of architecture view.

Note: An architecture viewpoint can also be seen as 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.

In some sections of this standard, the term "viewpoint" is used as a synonym for "architecture viewpoint".

See also Section A.38.

3.19 Architecture Vision

A succinct description of the Target Architecture that describes its business value and the changes to the enterprise that will result from its successful deployment. It serves as an aspirational vision and a boundary for detailed architecture development.

Note: Phase A (Architecture Vision) is described in Part II, Chapter 6.

3.20 Artifact

An architectural work product that describes an aspect of the architecture.

See also Section 3.23.

3.21 Baseline

A specification that has been formally reviewed and agreed upon, that thereafter serves as the basis for further development or change and that can be changed only through formal change control procedures or a type of procedure such as configuration management.

3.22 Boundaryless Information Flow™

A shorthand representation of "access to integrated information to support business process improvements" representing a desired state of an enterprise's infrastructure specific to the business needs of the organization.

Note: The need for Boundaryless Information Flow — a trademark of The Open Group — is described in the TOGAF® Series Guide: The TOGAF Integrated Information Infrastructure Reference Model (III-RM).

3.23 Building Block

A (potentially re-usable) component of enterprise capability that can be combined with other building blocks to deliver architectures and solutions.

Note: 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".

Building blocks are described in Part IV, Chapter 33.

See also Section 3.20.

3.24 Business Architecture

A representation of holistic, multi-dimensional business views of: capabilities, end-to-end value delivery, information, and organizational structure; and the relationships among these business views and strategies, products, policies, initiatives, and stakeholders.

Note: Business Architecture relates business elements to business goals and elements of other domains.

Business Architecture is described in Part II, Chapter 7.

3.25 Business Capability

A particular ability that a business may possess or exchange to achieve a specific purpose.

3.26 Business Function

Delivers business capabilities closely aligned to an organization, but not necessarily explicitly governed by the organization.

3.27 Business Governance

Concerned with ensuring that the business processes and policies (and their operation) deliver the business outcomes and adhere to relevant business regulation.

3.28 Business Model

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

3.29 Business Service

Supports business capabilities through an explicitly defined interface and is explicitly governed by an organization.

3.30 Capability

An ability that an organization, person, or system possesses.

Note: For example, Enterprise Architecture, marketing, customer contact, or outbound telemarketing.

3.31 Capability Architecture

A highly detailed description of the architectural approach to realize a particular solution or solution aspect.

3.32 Capability Increment

A discrete portion of a capability architecture that delivers specific value. When all increments have been completed, the capability has been realized.

3.33 Communications and Stakeholder Management

The management of needs of stakeholders of the Enterprise Architecture practice. It also manages the execution of communication between the practice and the stakeholders and the practice and the consumers of its services.

Note: Architecture stakeholder management is described in Chapter 21.

3.34 Concern

An interest in a system relevant to one or more of its stakeholders.

Note: 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.

See also Section 3.72.

3.35 Course of Action

Direction and focus provided by strategic goals and objectives, often to deliver the value proposition characterized in the business model.

3.36 Data Architecture

A description of the structure and interaction of the enterprise's major types and sources of data, logical data assets, physical data assets, and data management resources.

Note: Data Architecture is described in Part II, Chapter 9.

3.37 Deliverable

An architectural work product that is contractually specified and in turn formally reviewed, agreed, and signed off by the stakeholders.

Note: 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.

3.38 Enterprise

The highest level (typically) of description of an organization and typically covers all missions and functions. An enterprise will often span multiple organizations.

3.39 Enterprise Continuum

A categorization mechanism useful 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.

See also Section 3.9 and Section 3.71.

3.40 Foundation Architecture

Generic building blocks, their inter-relationships with other building blocks, combined with the principles and guidelines that provide a foundation on which more specific architectures can be built.

3.41 Framework

A structure for content or process that can be used as a tool to structure thinking, ensuring consistency and completeness.

3.42 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 Part III, Chapter 23.

3.43 Governance

The discipline of monitoring, managing, and steering a business (or IS/IT landscape) to deliver the business outcome required.

See also Section 3.13, Section 3.27, and Section A.40 in Appendix A.

3.44 Information

Any communication or representation of facts, data, or opinions, in any medium or form, including textual, numerical, graphic, cartographic, narrative, or audio-visual forms.

3.45 Information System Service

1. A discrete behavior requestable from an application (e.g., log in, book train seat, transfer money).

   Note: It supports and enables business roles and processes by capturing or providing data or automating a process. It can be coarse-grained or fine-grained (cf. a use-case or user story). It can be found in and invoked via an interface.

2. The automated elements of a business service.

3.46 Information Technology (IT)

1. The lifecycle management of information and related technology used by an organization.
2. An umbrella term that includes all or some of the subject areas relating to the computer industry, such as Business Continuity, Business IT Interface, Business Process Modeling and Management, Communication, Compliance and Legislation, Computers, Content Management, Hardware, Information Management, Internet, Offshoring, Networking, Programming and Software, Professional Issues, Project Management, Security,

Standards, Storage, Voice and Data Communications. Various countries and industries employ other umbrella terms to describe this same collection.

3. A term commonly assigned to a department within an organization tasked with provisioning some or all of the domains described in (2) above.
4. Alternate names commonly adopted include Information Services, Information Management, et al.

3.47 Interoperability

1. The ability to share information and services.
2. The ability of two or more systems or components to exchange and use information.
3. The ability of systems to provide and receive services from other systems and to use the services so interchanged to enable them to operate effectively together.

3.48 Logical

An implementation-independent definition of the architecture, often grouping related physical entities according to their purpose and structure.

Note: For example, the products from multiple infrastructure software vendors can all be logically grouped as Java® application server platforms.

3.49 Metadata

Data about data, of any sort in any media, that describes the characteristics of an entity.

3.50 Metamodel

A model that describes how and with what the architecture will be described in a structured way.

3.51 Method

A defined, repeatable approach to address a particular type of problem.

3.52 Modeling

A technique through construction of models which enables a subject to be represented in a form that enables reasoning, insight, and clarity concerning the essence of the subject matter.

3.53 Model Kind

Conventions for a type of modeling.

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

3.54 Objective

A time-bounded milestone for an organization used to demonstrate progress towards a goal; for example, "Increase capacity utilization by 30% by the end of 2019 to support the planned increase in market share".

3.55 Organization Map

An articulation of the relationships between the primary entities that make up the enterprise, its partners, and stakeholders.

3.56 Pattern

A technique for putting building blocks into context; for example, to describe a re-usable solution to a problem.

Note: Building blocks are what you use: (architecture) patterns can tell you how you use them, when, why, and what trade-offs you have to make in doing so.

See also Section 3.23.

3.57 Physical

A description of a real-world entity. Physical elements in an Enterprise Architecture may still be considerably abstracted from Solution Architecture, design, or implementation views.

3.58 Principle

See Section 3.16.

3.59 Reference Model (RM)

An abstract framework for understanding significant relationships among the entities of [an] environment, and for the development of consistent standards or specifications supporting that environment.

Note: A reference model is based on a small number of unifying concepts and may be used as a basis for education and explaining standards to a non-specialist. A reference model is not directly tied to any standards, technologies, or other concrete implementation details, but it does seek to provide common semantics that can be used unambiguously across and between different implementations.

Source: OASIS®; refer to www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-rm.

3.60 Repository

A system that manages all of the data of an enterprise, including data and process models and other enterprise information.

Note: The data in a repository is much more extensive than that in a data dictionary, which generally defines only the data making up a database.

3.61 Requirement

A statement of need that must be met by a particular architecture or work package.

3.62 Roadmap

An abstracted plan for business or technology change, typically operating across multiple disciplines over multiple years. Normally used in the phrases Technology Roadmap, Architecture Roadmap, etc.

3.63 Role

1. The usual or expected function of an actor, or the part somebody or something plays in a particular action or event. An actor may have a number of roles.
2. The part an individual plays in an organization and the contribution they make through the application of their skills, knowledge, experience, and abilities.

See also Section 3.2.

3.64 Segment Architecture

A detailed, formal description of areas within an enterprise, used at the program or portfolio level to organize and align change activity.

See also Section 3.74.

3.65 Service

1. A repeatable activity; a discrete behavior that a building block may be requested or otherwise triggered to perform.

   Note: Examples include check customer credit, provide weather data, and consolidate drilling reports. It serves a client or customer by delivering an output or changing system state. It can be defined in a logical service contract that defines input and output flows and/or state changes. It encapsulates any building block that processes the input and output flows. It may be one of several services in a service portfolio or Service-Level Agreement (SLA). It may be invoked via an interface. It can be coarse-grained (build a house) or fine-grained (retrieve an address).

2. An element of behavior that provides specific functionality in response to requests from actors or other services.

3.66 Service Orientation

Viewing an enterprise, system, or building block in terms of services provided and consumed.

See also Section 3.67.

3.67 Service-Oriented Architecture (SOA)

An architectural style that supports service orientation.

See also Section 3.6 and Section 3.66.

3.68 Service Portfolio

A collection of services, potentially an interface definition.

Note: It is used in the TOGAF framework to define the requirement for a building block or system.

3.69 Solution Architecture

A description of a discrete and focused business operation or activity and how IS/IT supports that operation.

Note: A Solution Architecture typically applies to a single project or project release, assisting in the translation of requirements into a solution vision, high-level business and/or IT system specifications, and a portfolio of implementation tasks.

3.70 Solution Building Block (SBB)

A candidate solution which conforms to the specification of an Architecture Building Block (ABB).

3.71 Solutions Continuum

A part of the Enterprise Continuum. A repository of re-usable solutions for future implementation efforts. It contains implementations of the corresponding definitions in the Architecture Continuum.

See also Section 3.39 and Section 3.9.

3.72 Stakeholder

An individual, team, organization, or class thereof, having an interest in a system.

3.73 Standards Information Base (SIB)

A database of standards that can be used to define the particular services and other components of an Organization-Specific Architecture.

Note: The Standards Information Base is described in Part V, Section 37.4.

3.74 Strategic Architecture

A summary formal description of the enterprise, providing an organizing framework for operational and change activity, and an executive-level, long-term view for direction setting.

3.75 Target Architecture

The description of a future state of the architecture being developed for an organization.

Note: There may be several future states developed as a roadmap to show the evolution of the architecture to a target state.

3.76 Taxonomy of Architecture Views

The organized collection of all architecture views pertinent to an architecture.

3.77 Technology Architecture

A description of the structure and interaction of the technology services and technology components.

Note: Technology Architecture is described in Part II, Chapter 11.

3.78 Technology Component

1. A technology building block. A generic infrastructure technology that supports and enables application or data components (directly or indirectly) by providing technology services.
2. An encapsulation of technology infrastructure that represents a class of technology product or specific technology product.

3.79 Technology Service

A technical capability required to provide enabling infrastructure that supports the delivery of applications.

3.80 Transition Architecture

A formal description of one state of the architecture at an architecturally significant point in time.

Note: One or more Transition Architectures may be used to describe the progression in time from the Baseline to the Target Architecture.

Transition Architecture is described in Part IV, Section 32.2.3.

3.81 Value Stream

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

3.82 View

See Section 3.17.

3.83 Viewpoint

See Section 3.18.

3.84 Viewpoint Library

A collection of the specifications of architecture viewpoints contained in the Reference Library portion of the Architecture Repository.

3.85 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.

The TOGAF Standard, Version 9.2

Part II:

Architecture Development Method (ADM)

The Open Group

Chapter 4

Introduction to Part II

This chapter describes the Architecture Development Method (ADM) cycle, adapting the ADM, architecture scope, and architecture integration.

4.1 ADM Overview

The TOGAF ADM is the result of continuous contributions from a large number of architecture practitioners. It describes a method for developing and managing the lifecycle of an Enterprise Architecture, and forms the core of the TOGAF standard. It integrates elements of the TOGAF standard described in this document as well as other available architectural assets, to meet the business and IT needs of an organization.

4.1.1 The ADM, Enterprise Continuum, and Architecture Repository

The Enterprise Continuum provides a framework and context to support the leverage of relevant architecture assets in executing the ADM. These assets may include Architecture Descriptions, models, and patterns taken from a variety of sources, as explained in Part V: Enterprise Continuum & Tools.

The Enterprise Continuum categorizes architectural source material — both the contents of the organization's own enterprise repositories and the set of relevant, available reference models and standards in the industry.

The practical implementation of the Enterprise Continuum will typically take the form of an Architecture Repository (see Part V, Chapter 37) that includes reference architectures, models, and patterns that have been accepted for use within the enterprise, and actual architectural work done previously within the enterprise. The architect would seek to re-use as much as possible from the Architecture Repository that was relevant to the project at hand. (In addition to the collection of architecture source material, the repository would also contain architecture development work-in-progress.)

At relevant places throughout the ADM there are reminders to consider which, if any, architecture assets from the Architecture Repository the architect should use. In some cases — for example, in the development of a Technology Architecture — this may be the TOGAF Foundation Architecture. In other cases — for example, in the development of a Business Architecture — it may be a reference model for e-Commerce taken from the industry at large.

The criteria for including source materials in an organization's Architecture Repository will typically form part of the Enterprise Architecture Governance process. These governance processes should consider available resources both within and outside the enterprise in order to determine when general resources can be adapted for specific enterprise needs and also to determine where specific solutions can be generalized to support wider re-use.

While using the ADM, the architect is developing a snapshot of the enterprise's decisions and their implications at particular points in time. Each iteration of the ADM will populate an organization-specific landscape with all the architecture assets identified and leveraged through the process, including the final organization-specific architecture delivered.

Architecture development is a continuous, cyclical process, and in executing the ADM repeatedly over time, the architect gradually adds more and more content to the organization's Architecture Repository. Although the primary focus of the ADM is on the development of the enterprise-specific architecture, in this wider context the ADM can also be viewed as the process of populating the enterprise's own Architecture Repository with relevant re-usable building blocks taken from the "left", more generic side of the Enterprise Continuum.

In fact, the first execution of the ADM will often be the hardest, since the architecture assets available for re-use will be relatively scarce. Even at this stage of development, however, there will be architecture assets available from external sources such as the TOGAF standard, as well as the IT industry at large, that could be leveraged in support of the effort.

Subsequent executions will be easier, as more and more architecture assets become identified, are used to populate the organization's Architecture Repository, and are thus available for future re-use.

4.1.2 The ADM and the Foundation Architecture

The ADM is also useful to populate the Foundation Architecture of an enterprise. Business requirements of an enterprise may be used to identify the necessary definitions and selections in the Foundation Architecture. This could be a set of re-usable common models, policy and governance definitions, or even as specific as overriding technology selections (e.g., if mandated by law). Population of the Foundation Architecture follows similar principles as for an Enterprise Architecture, with the difference that requirements for a whole enterprise are restricted to the overall concerns and thus less complete than for a specific enterprise.

It is important to recognize that existing models from these various sources, when integrated, may not necessarily result in a coherent Enterprise Architecture. "Integratability" of Architecture Descriptions is considered in Section 4.6.

4.1.3 ADM and Supporting Guidelines and Techniques

The application of the TOGAF ADM is supported by an extended set of resources — guidelines, templates, checklists, and other detailed materials. These are included in:

- Part III: ADM Guidelines & Techniques
- White Papers and Guides published by The Open Group, classified and referenced in the TOGAF Library (see https://publications.opengroup.org/togaf-library)

The individual guidelines and techniques are described separately, so that they can be referenced from the relevant points in the ADM as necessary, rather than having the detailed text clutter the description of the ADM itself.

4.2 Architecture Development Cycle

4.2.1 Key Points

The following are the key points about the ADM:

- The ADM is iterative, over the whole process, between phases, and within phases (see Part III, Chapter 18)

  For each iteration of the ADM, a fresh decision must be taken as to:

  — The breadth of coverage of the enterprise to be defined

  — The level of detail to be defined

  — The extent of the time period aimed at, including the number and extent of any intermediate time periods

  — The architectural assets to be leveraged, including:

    — Assets created in previous iterations of the ADM cycle within the enterprise

    — Assets available elsewhere in the industry (other frameworks, systems models, vertical industry models, etc.)

- These decisions should be based on a practical assessment of resource and competence availability, and the value that can realistically be expected to accrue to the enterprise from the chosen scope of the architecture work

- As a generic method, the ADM is intended to be used by enterprises in a wide variety of different geographies and applied in different vertical sectors/industry types

  As such, it may be, but does not necessarily have to be, tailored to specific needs. For example, it may be used in conjunction with the set of deliverables of another framework, where these have been deemed to be more appropriate for a specific organization. (For example, many US Federal agencies have developed individual frameworks that define the deliverables specific to their particular departmental needs.)

These issues are considered in detail in Section 4.3.

4.2.2 Basic Structure

The basic structure of the ADM is shown in Figure 4-1.

Throughout the ADM cycle, there needs to be frequent validation of results against the original expectations, both those for the whole ADM cycle, and those for the particular phase of the process.

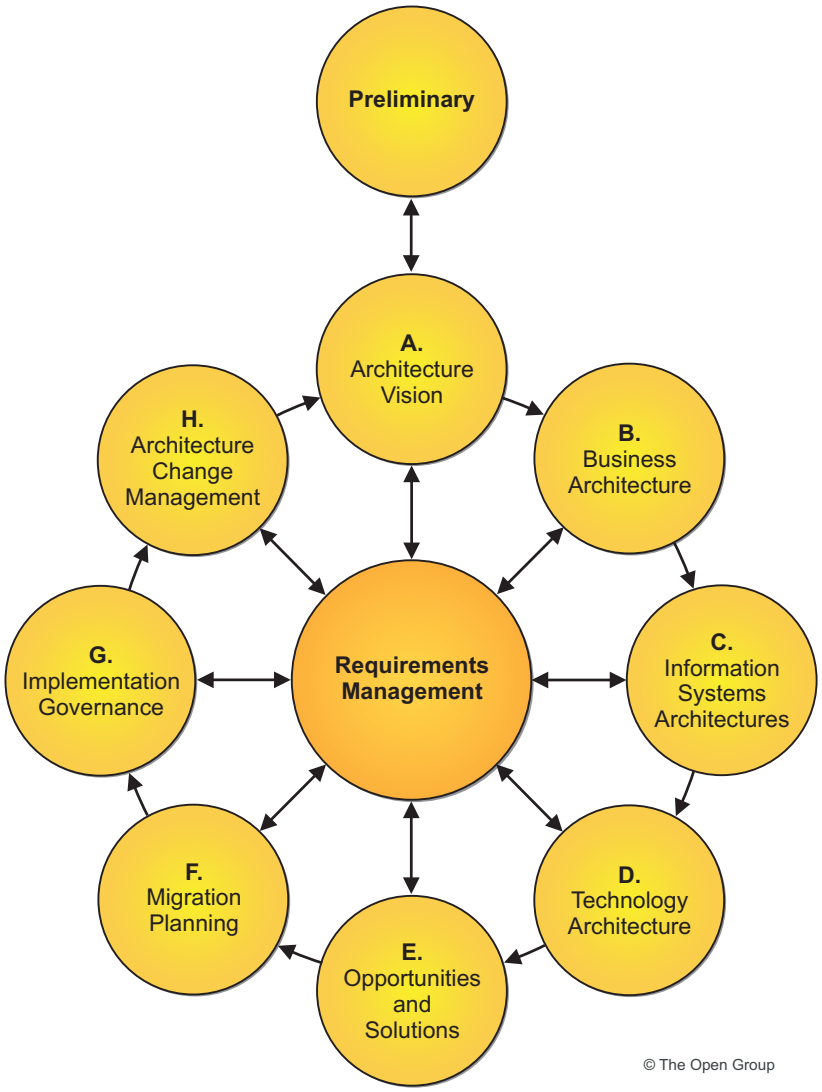

Figure 4-1 Architecture Development Cycle

The phases of the ADM cycle are further divided into steps; for example, the steps within the architecture development phases (B, C, D) are as follows:

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

- Finalize the Architecture
- Create the Architecture Definition Document

The Requirements Management phase is a continuous phase which ensures that any changes to requirements are handled through appropriate governance processes and reflected in all other phases.

An enterprise may choose to record all new requirements, including those which are in scope of the current Statement of Architecture Work through a single Requirements Repository.

The phases of the cycle are described in detail in the following chapters within Part II.

Note that output is generated throughout the process, and that the output in an early phase may be modified in a later phase. The versioning of output is managed through version numbers. In all cases, the ADM numbering scheme is provided as an example. It should be adapted by the architect to meet the requirements of the organization and to work with the architecture tools and repositories employed by the organization.

In particular, a version numbering convention is used within the ADM to illustrate the evolution of Baseline and Target Architecture Definitions. Table 4-1 describes how this convention is used.

Phase	Deliverable	Content	Version	Description
A: Architecture Vision	Architecture Vision	Business Architecture	0.1	Version 0.1 indicates that a high-level outline of the architecture is in place.
		Data Architecture	0.1	Version 0.1 indicates that a high-level outline of the architecture is in place.
		Application Architecture	0.1	Version 0.1 indicates that a high-level outline of the architecture is in place.
		Technology Architecture	0.1	Version 0.1 indicates that a high-level outline of the architecture is in place.
B: Business Architecture	Architecture Definition Document	Business Architecture	1.0	Version 1.0 indicates a formally reviewed, detailed architecture.
C: Information Systems Architecture	Architecture Definition Document	Data Architecture	1.0	Version 1.0 indicates a formally reviewed, detailed architecture.
		Application Architecture	1.0	Version 1.0 indicates a formally reviewed, detailed architecture.
D: Technology Architecture	Architecture Definition Document	Technology Architecture	1.0	Version 1.0 indicates a formally reviewed, detailed architecture.

Table 4-1 ADM Version Numbering Convention

4.3 Adapting the ADM

The ADM is a generic method for architecture development, which is designed to deal with most system and organizational requirements. However, it will often be necessary to modify or extend the ADM to suit specific needs. One of the tasks before applying the ADM is to review its components for applicability, and then tailor them as appropriate to the circumstances of the individual enterprise. This activity may well produce an "enterprise-specific" ADM.

One reason for wanting to adapt the ADM, which it is important to stress, is that the order of the phases in the ADM is to some extent dependent on the maturity of the architecture discipline within the enterprise. For example, if the business case for doing architecture at all is not well recognized, then creating an Architecture Vision is almost always essential; and a detailed Business Architecture often needs to come next, in order to underpin the Architecture Vision, detail the business case for remaining architecture work, and secure the active participation of key stakeholders in that work. In other cases a slightly different order may be preferred; for example, a detailed inventory of the baseline environment may be done before undertaking the Business Architecture.

The order of phases may also be defined by the Architecture Principles and business principles of an enterprise. For example, the business principles may dictate that the enterprise be prepared to adjust its business processes to meet the needs of a packaged solution, so that it can be implemented quickly to enable fast response to market changes. In such a case, the Business Architecture (or at least the completion of it) may well follow completion of the Information Systems Architecture or the Technology Architecture.

Another reason for wanting to adapt the ADM is if the TOGAF framework is to be integrated with another enterprise framework (as explained in Part I, Section 2.10). For example, an enterprise may wish to use the TOGAF framework and its generic ADM in conjunction with the Zachman Framework, or another Enterprise Architecture framework that has a defined set of deliverables specific to a particular vertical sector: Government, Defense, e-Business, Telecommunications, etc. The ADM has been specifically designed with this potential integration in mind.

Other possible reasons for wanting to adapt the ADM include:

- The ADM is one of the many corporate processes that make up the corporate governance model

  It is complementary to, and supportive of, other standard program management processes, such as those for authorization, risk management, business planning and budgeting, development planning, systems development, and procurement.

- The ADM is being mandated for use by a prime or lead contractor in an outsourcing situation, and needs to be tailored to achieve a suitable compromise between the contractor's existing practices and the contracting enterprise's requirements

- The enterprise is a small-to-medium enterprise, and wishes to use a "cut-down" method more attuned to the reduced level of resources and system complexity typical of such an environment

- The enterprise is very large and complex, comprising many separate but interlinked "enterprises" within an overall collaborative business framework, and the architecture method needs to be adapted to recognize this

  Different approaches to planning and integration may be used in such cases, including the following (possibly in combination):

— Top-down planning and development — designing the whole interconnected meta-enterprise as a single entity (an exercise that typically stretches the limits of practicality)

— Development of a "generic" or "reference" architecture, typical of the enterprises within the organization, but not representing any specific enterprise, which individual enterprises are then expected to adapt in order to produce an architecture "instance" suited to the particular enterprise concerned

— Replication — developing a specific architecture for one enterprise, implementing it as a proof-of-concept, and then taking that as a "reference architecture" to be cloned in other enterprises

- In a vendor or production environment, a generic architecture for a family of related products is often referred to as a "Product Line Architecture", and the analogous process to that outlined above is termed "(Architecture-based) Product Line Engineering". The ADM is targeted primarily at architects in IT user enterprises, but a vendor organization whose products are IT-based might well wish to adapt it as a generic method for a Product Line Architecture development.

4.4 Architecture Governance

The ADM, whether adapted by the organization or used as documented here, is a key process to be managed in the same manner as other architecture artifacts classified through the Enterprise Continuum and held in the Architecture Repository. The Architecture Board should be satisfied that the method is being applied correctly across all phases of an architecture development iteration. Compliance with the ADM is fundamental to the governance of the architecture, to ensure that all considerations are made and all required deliverables are produced.

The management of all architectural artifacts, governance, and related processes should be supported by a controlled environment. Typically, this would be based on one or more repositories supporting versioned objects, process control, and status.

The major information areas managed by a governance repository should contain the following types of information:

- **Reference Data** (collateral from the organization's own repositories/Enterprise Continuum, including external data; e.g., COBIT, the IT4IT Reference Architecture): used for guidance and instruction during project implementation

  This includes the details of information outlined above. The reference data includes a description of the governance procedures themselves.

- **Process Status**: all information regarding the state of any governance processes will be managed

  Examples of this include outstanding compliance requests, dispensation requests, and compliance assessments investigations.

- **Audit Information**: this will record all completed governance process actions and will be used to support:

  — Key decisions and responsible personnel for any architecture project that has been sanctioned by the governance process

  — A reference for future architectural and supporting process developments, guidance, and precedence

The governance artifacts and process are themselves part of the contents of the Architecture Repository.

4.5 Scoping the Architecture

There are many reasons to constrain (or restrict) the scope of the architectural activity to be undertaken, most of which relate to limits in:

- The organizational authority of the team producing the architecture
- The objectives and stakeholder concerns to be addressed within the architecture
- The availability of people, finance, and other resources

The scope chosen for the architecture activity should ideally allow the work of all architects within the enterprise to be effectively governed and integrated. This requires a set of aligned "architecture partitions" that ensure architects are not working on duplicate or conflicting activities. It also requires the definition of re-use and compliance relationships between architecture partitions.

The division of the enterprise and its architecture-related activity is discussed in more detail in Chapter 36.

Four dimensions are typically used in order to define and limit the scope of an architecture:

- **Breadth**: what is the full extent of the enterprise, and what part of that extent will this architecting effort deal with?
  - Many enterprises are very large, effectively comprising a federation of organizational units that could validly be considered enterprises in their own right
  - The modern enterprise increasingly extends beyond its traditional boundaries, to embrace a fuzzy combination of traditional business enterprise combined with suppliers, customers, and partners
- **Depth**: to what level of detail should the architecting effort go?

  How much architecture is "enough"? What is the appropriate demarcation between the architecture effort and other, related activities (system design, system engineering, system development)?
- **Time Period**: what is the time period that needs to be articulated for the Architecture Vision, and does it make sense (in terms of practicality and resources) for the same period to be covered in the detailed Architecture Description?

  If not, how many Transition Architectures are to be defined, and what are their time periods?
- **Architecture Domains**: a complete Enterprise Architecture description should contain all four architecture domains (business, data, application, technology), but the realities of resource and time constraints often mean there is not enough time, funding, or resources to build a top-down, all-inclusive Architecture Description encompassing all four architecture domains, even if the enterprise scope is chosen to be less than the full extent of the overall enterprise

Typically, the scope of an architecture is first expressed in terms of breadth, depth, and time. Once these dimensions are understood, a suitable combination of architecture domains can be selected that are appropriate to the problem being addressed. Techniques for using the ADM to develop a number of related architectures are discussed in Chapter 19.

The four dimensions of architecture scope are explored in detail below. In each case, particularly in largescale environments where architectures are necessarily developed in a federated manner, there is a danger of architects optimizing within their own scope of activity, instead of at the level of the overall enterprise. It is often necessary to sub-optimize in a particular area, in order to optimize at the enterprise level. The aim should always be to seek the highest level of commonality and focus on scalable and re-usable modules in order to maximize re-use at the enterprise level.

4.5.1 Breadth

One of the key decisions is the focus of the architecture effort, in terms of the breadth of overall enterprise activity to be covered (which specific business sectors, functions, organizations, geographical areas, etc.).

It is often necessary to have a number of different architectures existing across an enterprise, focused on particular timeframes, business functions, or business requirements.

For large complex enterprises, federated architectures — independently developed, maintained, and managed architectures that are subsequently integrated within an integration framework — are typical. Such a framework specifies the principles for interoperability, migration, and conformance. This allows specific business units to have architectures developed and governed as stand-alone architecture projects. More details and guidance on specifying the interoperability requirements for different solutions can be found in Part III, Chapter 25.

The feasibility of a single enterprise-wide architecture for every business function or purpose may be rejected as too complex and unwieldy. In these circumstances it is suggested that a number of different Enterprise Architectures exist across an enterprise. These Enterprise Architectures focus on particular timeframes, business segments or functions, and specific organizational requirements. In such a case we need to create the overarching Enterprise Architecture as a "federation" of these Enterprise Architectures. An effective way of managing and exploiting these Enterprise Architectures is to adopt a publish-and-subscribe model that allows architecture to be brought under a governance framework. In such a model, architecture developers and architecture consumers in projects (the supply and demand sides of architecture work) sign up to a mutually beneficial framework of governance that ensures that:

- Architectural material is of good quality, up-to-date, fit-for-purpose, and published (reviewed and agreed to be made public)
- Usage of architecture material can be monitored, and compliance with standards, models, and principles can be exhibited, via:
  - A Compliance Assessment process that describes what the user is subscribing to, and assesses their level of compliance
  - A dispensation process that may grant dispensations from adherence to architecture standards and guidelines in specific cases (usually with a strong business imperative)

Publish and subscribe techniques are being developed as part of general IT governance and specifically for the Defense sphere.

4.5.2 Depth

Care should be taken to judge the appropriate level of detail to be captured, based on the intended use of the Enterprise Architecture and the decisions to be made based on it. It is important that a consistent and equal level of depth be completed in each architecture domain (business, data, application, technology) included in the architecture effort. If pertinent detail is omitted, the architecture may not be useful. If unnecessary detail is included, the architecture effort may exceed the time and resources available, and/or the resultant architecture may be confusing or cluttered. Developing architectures at different levels of detail within an enterprise is discussed in more detail in Chapter 19.

It is also important to predict the future uses of the architecture so that, within resource limitations, the architecture can be structured to accommodate future tailoring, extension, or re-use. The depth and detail of the Enterprise Architecture needs to be sufficient for its purpose, and no more.

Iterations of the ADM will build on the artifacts and the capabilities created during previous iterations.

There is a need to document all the models in an enterprise, to the level of detail appropriate to the need of the current ADM cycle. The key is to understand the status of the enterprise's architecture work, and what can realistically be achieved with the resources and competencies available, and then focus on identifying and delivering the value that is achievable. Stakeholder value is a key focus: too broad a scope may deter some stakeholders (no return on investment).

4.5.3 Time Period

The ADM is described in terms of a single cycle of Architecture Vision, and a set of Target Architectures (Business, Data, Application, Technology) that enable the implementation of the vision.

In such cases, a wider view may be taken, whereby an enterprise is represented by several different architecture instances (for example, strategic, segment, capability), each representing the enterprise at a particular point in time. One architecture instance will represent the current enterprise state (the "as-is", or baseline). Another architecture instance, perhaps defined only partially, will represent the ultimate target end-state (the "vision"). In-between, intermediate or "Transition Architecture" instances may be defined, each comprising its own set of Target Architecture Descriptions. An example of how this might be achieved is given in Part III, Chapter 19.

By this approach, the Target Architecture work is split into two or more discrete stages:

1. First, develop Target Architecture Descriptions for the overall (largescale) system, demonstrating a response to stakeholder objectives and concerns for a relatively distant timeframe (for example, a six-year period).
2. Then develop one or more "Transition Architecture" descriptions, as increments or plateaus, each in line with and converging on the Target Architecture Descriptions, and describing the specifics of the increment concerned.

In such an approach, the Target Architectures are evolutionary in nature, and require periodic review and update according to evolving business requirements and developments in technology, whereas the Transition Architectures are (by design) incremental in nature, and in principle should not evolve during the implementation phase of the increment, in order to avoid the "moving target" syndrome. This, of course, is only possible if the implementation schedule is under tight control and relatively short (typically less than two years).

The Target Architectures remain relatively generic, and because of that are less vulnerable to obsolescence than the Transition Architectures. They embody only the key strategic architectural decisions, which should be blessed by the stakeholders from the outset, whereas the detailed architectural decisions in the Transition Architectures are deliberately postponed as far as possible (i.e., just before implementation) in order to improve responsiveness *vis a vis* new technologies and products.

The enterprise evolves by migrating to each of these Transition Architectures in turn. As each Transition Architecture is implemented, the enterprise achieves a consistent, operational state on the way to the ultimate vision. However, this vision itself is periodically updated to reflect changes in the business and technology environment, and in effect may never actually be achieved, as originally described. The whole process continues for as long as the enterprise exists and continues to change.

Such a breakdown of the Architecture Description into a family of related architecture products of course requires effective management of the set and their relationships.

4.5.4 Architecture Domains

A complete Enterprise Architecture should address all four architecture domains (business, data, application, technology), but the realities of resource and time constraints often mean there is not enough time, funding, or resources to build a top-down, all-inclusive Architecture Description encompassing all four architecture domains.

Architecture descriptions will normally be built with a specific purpose in mind — a specific set of business drivers that drive the architecture development — and clarifying the specific issue(s) that the Architecture Description is intended to help explore, and the questions it is expected to help answer, is an important part of the initial phase of the ADM.

For example, if the purpose of a particular architecture effort is to define and examine technology options for achieving a particular capability, and the fundamental business processes are not open to modification, then a full Business Architecture may well not be warranted. However, because the Data, Application, and Technology Architectures build on the Business Architecture, the Business Architecture still needs to be thought through and understood.

While circumstances may sometimes dictate building an Architecture Description not containing all four architecture domains, it should be understood that such an architecture cannot, by definition, be a complete Enterprise Architecture. One of the risks is lack of consistency and therefore ability to integrate. Integration either needs to come later — with its own costs and risks — or the risks and trade-offs involved in not developing a complete and integrated architecture need to be articulated by the architect, and communicated to and understood by the enterprise management.

4.6 Architecture Integration

Architectures that are created to address a subset of issues within an enterprise require a consistent frame of reference so that they can be considered as a group as well as point deliverables. The dimensions that are used to define the scope boundary of a single architecture (e.g., level of detail, architecture domain, etc.) are typically the same dimensions that must be addressed when considering the integration of many architectures. Figure 4-2 illustrates how different types of architecture need to co-exist.

At the present time, the state of the art is such that architecture integration can be accomplished only at the lower end of the integratability spectrum. Key factors to consider are the granularity

and level of detail in each artifact, and the maturity of standards for the interchange of architectural descriptions.

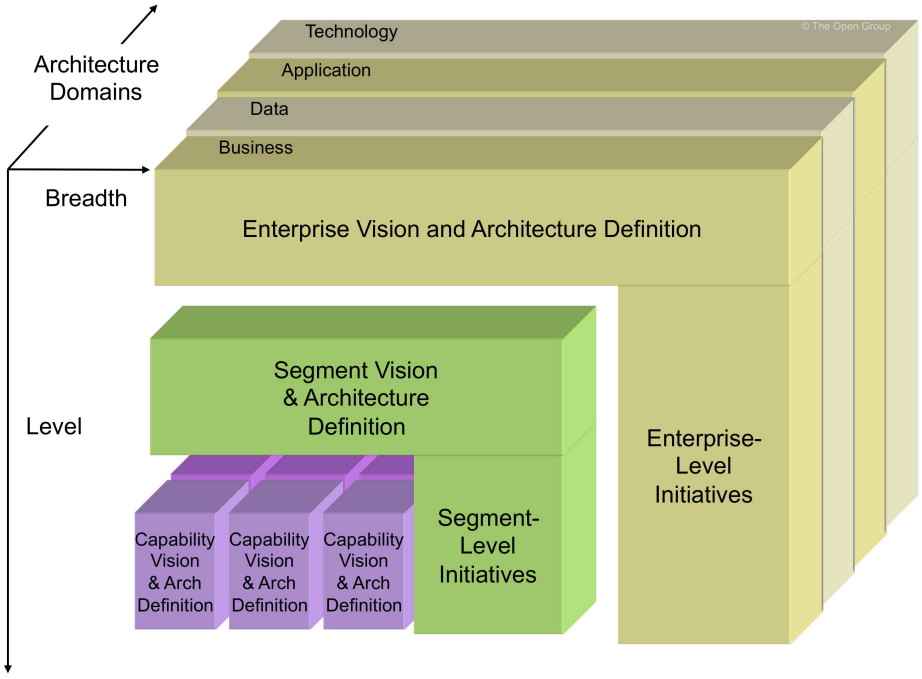

Figure 4-2 Integration of Architecture Artifacts

As organizations address common themes (such as Service-Oriented Architecture (SOA), and integrated information infrastructure), and universal data models and standard data structures emerge, integration toward the high end of the spectrum will be facilitated. However, there will always be the need for effective standards governance to reduce the need for manual co-ordination and conflict resolution.

4.7 Summary

The TOGAF ADM defines a recommended sequence for the various phases and steps involved in developing an architecture, but it cannot recommend a scope — this has to be determined by the organization itself, bearing in mind that the recommended sequence of development in the ADM process is an iterative one, with the depth and breadth of scope and deliverables increasing with each iteration. Each iteration will add resources to the organization's Architecture Repository.

While a complete framework is useful (indeed, essential) to have in mind as the ultimate long-term goal, in practice there is a key decision to be made as to the scope of a specific Enterprise Architecture effort. This being the case, it is vital to understand the basis on which scoping decisions are being made, and to set expectations right for what is the goal of the effort.

The main guideline is to focus on what creates value to the enterprise, and to select horizontal and vertical scope, and time periods, accordingly. Whether or not this is the first time around, understand that this exercise will be repeated, and that future iterations will build on what is being created in the current effort, adding greater width and depth.

Chapter 5

Preliminary Phase

This chapter describes the preparation and initiation activities required to meet the business directive for a new Enterprise Architecture, including the definition of an Organization-Specific Architecture framework and the definition of principles.

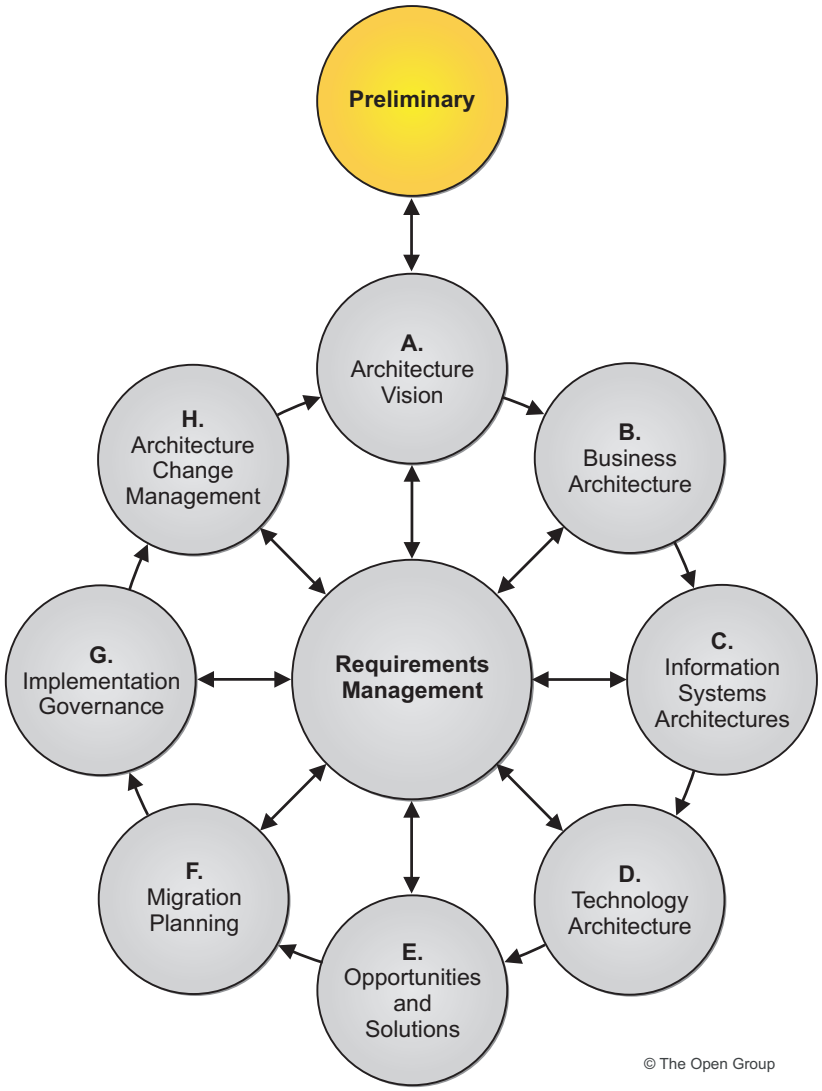

Figure 5-1 Preliminary Phase

5.1 Objectives

The objectives of the Preliminary Phase are to:

1. Determine the Architecture Capability desired by the organization:
   - Review the organizational context for conducting Enterprise Architecture
   - Identify and scope the elements of the enterprise organizations affected by the Architecture Capability
   - Identify the established frameworks, methods, and processes that intersect with the Architecture Capability
   - Establish Capability Maturity target
2. Establish the Architecture Capability:
   - Define and establish the Organizational Model for Enterprise Architecture
   - Define and establish the detailed process and resources for Architecture Governance
   - Select and implement tools that support the Architecture Capability
   - Define the Architecture Principles

5.2 Inputs

This section defines the inputs to the Preliminary Phase.

5.2.1 Reference Materials External to the Enterprise

- The TOGAF Library
- Other architecture framework(s), if required

5.2.2 Non-Architectural Inputs

- Board strategies and board business plans, business strategy, IT strategy, business principles, business goals, and business drivers, when pre-existing
- Major frameworks operating in the business; e.g., project/portfolio management
- Governance and legal frameworks, including Architecture Governance strategy, when pre-existing
- Architecture capability
- Partnership and contract agreements

5.2.3 Architectural Inputs

Pre-existing models for operating an Enterprise Architecture Capability can be used as a baseline for the Preliminary Phase. Inputs would include:

- Organizational Model for Enterprise Architecture (see Part IV, Section 32.2.16), including:
  - Scope of organizations impacted
  - Maturity assessment, gaps, and resolution approach
  - Roles and responsibilities for architecture team(s)
  - Budget requirements
  - Governance and support strategy
- Existing Architecture Framework, if any, including:
  - Architecture method
  - Architecture content
  - Configured and deployed tools
  - Architecture Principles
  - Architecture Repository

5.3 Steps

The TOGAF ADM is a generic method, intended to be used by a wide variety of different enterprises, and in conjunction with a wide variety of other architecture frameworks, if required. The Preliminary Phase therefore involves doing any necessary work to initiate and adapt the ADM to define an organization-specific framework. The issues involved with adapting the ADM to a specific organizational context are discussed in detail in Section 4.3.

The level of detail addressed in the Preliminary Phase will depend on the scope and goals of the overall architecture effort.

The order of the steps in the Preliminary Phase as well as the time at which they are formally started and completed should be adapted to the situation at hand in accordance with the established Architecture Governance.

The steps within the Preliminary Phase are as follows:

- Scope the enterprise organizations impacted (see Section 5.3.1)
- Confirm governance and support frameworks (see Section 5.3.2)
- Define and establish Enterprise Architecture team and organization (see Section 5.3.3)
- Identify and establish Architecture Principles (see Section 5.3.4)
- Tailor the TOGAF framework and, if any, other selected architecture frameworks (see Section 5.3.5)
- Develop a strategy and implementation plan for tools and techniques (see Section 5.3.6)

5.3.1 Scope the Enterprise Organizations Impacted

- Identify core enterprise (units) — those who are most affected and achieve most value from the work
- Identify soft enterprise (units) — those who will see change to their capability and work with core units but are otherwise not directly affected
- Identify extended enterprise (units) — those units outside the scoped enterprise who will be affected in their own Enterprise Architecture
- Identify communities involved (enterprises) — those stakeholders who will be affected and who are in groups of communities
- Identify governance involved, including legal frameworks and geographies (enterprises)

5.3.2 Confirm Governance and Support Frameworks

The architecture framework will form the keystone to the flavor (centralized or federated, light or heavy, etc.) of Architecture Governance organization and guidelines that need to be developed. Part of the major output of this phase is a framework for Architecture Governance. We need to understand how architectural material (standards, guidelines, models, compliance reports, etc.) is brought under governance; i.e., what type of governance repository characteristics are going to be required, what relationships and status recording are necessary to ascertain which governance process (dispensation, compliance, take-on, retirement, etc.) has ownership of an architectural artifact.

It is likely that the existing governance and support models of an organization will need to change to support the newly adopted architecture framework.

To manage the organizational change required to adopt the new architectural framework, the current enterprise governance and support models will need to be assessed to understand their overall shape and content. Additionally, the sponsors and stakeholders for architecture will need to be consulted on potential impacts that could occur.

Upon completion of this step, the architecture touch-points and likely impacts should be understood and agreed by relevant stakeholders.

5.3.3 Define and Establish Enterprise Architecture Team and Organization

- Determine existing enterprise and business capability
- Conduct an Enterprise Architecture/business change maturity assessment, if required
- Identify gaps in existing work areas
- Allocate key roles and responsibilities for Enterprise Architecture Capability management and governance
- Define requests for change to existing business programs and projects:
  - Inform existing Enterprise Architecture and IT architecture work of stakeholder requirements
  - Request assessment of impact on their plans and work
  - Identify common areas of interest

  - Identify any critical differences and conflicts of interest
  - Produce requests for change to stakeholder activities
- Determine constraints on Enterprise Architecture work
- Review and agree with sponsors and board
- Assess budget requirements

5.3.4 Identify and Establish Architecture Principles

Architecture Principles (see Part III, Chapter 20) are based on business principles and are critical in setting the foundation for Architecture Governance. Once the organizational context is understood, define a set of Architecture Principles that is appropriate to the enterprise.

5.3.5 Tailor the TOGAF Framework and, if any, Other Selected Architecture Framework(s)

In this step, determine what tailoring of the TOGAF framework is required. Consider the need for:

- **Terminology Tailoring**: architecture practitioners should use terminology that is generally understood across the enterprise

  Tailoring should produce an agreed terminology set for description of architectural content. Consideration should be given to the creation of an Enterprise Glossary, to be updated throughout the architecture process.

- **Process Tailoring**: the TOGAF ADM provides a generic process for carrying out architecture

  Process tailoring provides the opportunity to remove tasks that are already carried out elsewhere in the organization, add organization-specific tasks (such as specific checkpoints), and to align the ADM processes to external process frameworks and touch-points. Key touch-points to be addressed would include:

  - Links to (project and service) portfolio management processes
  - Links to project lifecycle
  - Links to operations handover processes
  - Links to operational management processes (including configuration management, change management, and service management)
  - Links to procurement processes

- **Content Tailoring**: using the TOGAF Architecture Content Framework and Enterprise Continuum as a basis, tailoring of content structure and classification approach allows adoption of third-party content frameworks and also allows for customization of the framework to support organization-specific requirements

5.3.6 Develop a Strategy and Implementation Plan for Tools and Techniques

There are many tools and techniques which may be used to develop Enterprise Architecture across many domains. The development of a tools strategy is recommended that reflects the understanding and level of formality required by the enterprise's stakeholders. Architecture content will be highly dependent on the scale, sophistication, and culture of both the stakeholders and the Architecture Capability within the organization. A tools strategy which recognizes the stakeholders' articulation requirements will enable more effective and rapid decision-making by stakeholders and their ownership of artifacts.

The strategy should encompass management techniques, decision management, workshop techniques, business modeling, detailed infrastructure modeling, office products, languages, and repository management as well as more formal architecture tools. For example, the Balanced Scorecard technique is a best practice performance measurement tool used by business schools and many organizations that can be used successfully in architecture projects.

The implementation of the tools strategy may be based on common desktop and office tools or may be based on a customized deployment of specialist management and architecture tools. Change management of the artifact deliverables is a major consideration and a degree of management control and governance of artifacts needs to be considered. Access to decisions needs to be managed carefully as many of the artifacts may contain sensitive information. Therefore the tools implementation, access, and security of the content needs to reflect the sensitivity requirements.

Issues in tools standardization are discussed in Part V, Chapter 38.

5.4 Outputs

The outputs of the Preliminary Phase may include, but are not restricted to:

- Organizational Model for Enterprise Architecture (see Part IV, Section 32.2.16), including:
  - 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
- Tailored Architecture Framework (see Part IV, Section 32.2.21), including:
  - Tailored architecture method
  - Tailored architecture content (deliverables and artifacts)
  - Architecture Principles (see Part IV, Section 32.2.4)
  - Configured and deployed tools
- Initial Architecture Repository (see Part IV, Section 32.2.5), populated with framework content
- Restatement of, or reference to, business principles, business goals, and business drivers (see Part IV, Section 32.2.9)

- Request for Architecture Work (optional) (see Part IV, Section 32.2.17)
- Architecture Governance Framework (see (Part VI, Section 44.2)

The outputs may include some or all of the following:

- Catalogs:
  - Principles catalog

5.5 Approach

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

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

The Enterprise Architecture provides a strategic, top-down view of an organization to enable executives, planners, architects, and engineers to coherently co-ordinate, integrate, and conduct their activities. The Enterprise Architecture framework provides the strategic context within which this team can operate.

Therefore, developing the Enterprise Architecture is not a solitary activity and the Enterprise Architects need to recognize the interoperability between their frameworks and the rest of the business.

Strategic, interim, and tactical business objectives and aspirations need to be met. Similarly, the Enterprise Architecture needs to reflect this requirement and allow for operation of architecture discipline at different levels within the organization.

Depending on the scale of the enterprise and the level of budgetary commitment to Enterprise Architecture discipline, a number of approaches may be adopted to sub-divide or partition architecture teams, processes, and deliverables. Approaches for architecture partitioning are discussed in Part V, Chapter 36. The Preliminary Phase should be used to determine the desired approach to partitioning and to establish the groundwork for the selected approach to be put into practice.

The Preliminary Phase may be revisited, from the Architecture Vision phase (see Part III, Chapter 18), in order to ensure that the organization's Architecture Capability is suitable to address a specific architecture problem.

5.5.1 Enterprise

One of the main challenges of Enterprise Architecture is that of enterprise scope.

The scope of the enterprise, and whether it is federated, will determine those stakeholders who will derive most benefit from the Enterprise Architecture Capability. It is imperative that a sponsor is appointed at this stage to ensure that the resultant activity has resources to proceed and the clear support of the business management. The enterprise may encompass many organizations and the duties of the sponsor are to ensure that all stakeholders are included in defining, establishing, and using the Architecture Capability.

5.5.2 Organizational Context

In order to make effective and informed decisions about the framework for architecture to be used within a particular enterprise, it is necessary to understand the context surrounding the architecture framework. Specific areas to consider would include:

- The commercial models for Enterprise Architecture and budgetary plans for Enterprise Architecture activity; where no such plans exist, the Preliminary Phase should be used to develop a budget plan
- The stakeholders for architecture in the enterprise; their key issues and concerns
- The intentions and culture of the organization, as captured within board business directives, business imperatives, business strategies, business principles, business goals, and business drivers
- Current processes that support execution of change and operation of the enterprise, including the structure of the process and also the level of rigor and formality applied within the organization

  Areas for focus should include:

  — Current methods for architecture description

  — Current project management frameworks and methods

  — Current systems management frameworks and methods

  — Current project portfolio management processes and methods

  — Current application portfolio management processes and methods

  — Current technology portfolio management processes and methods

  — Current information portfolio management processes and methods

  — Current systems design and development frameworks and methods
- The Baseline Architecture landscape, including the state of the enterprise and also how the landscape is currently represented in documentation form
- The skills and capabilities of the enterprise and specific organizations that will be adopting the framework

Review of the organizational context should provide valuable requirements on how to tailor the architecture framework in terms of:

- Level of formality and rigor to be applied

- Level of sophistication and expenditure required
- Touch-points with other organizations, processes, roles, and responsibilities
- Focus of content coverage

5.5.3 Requirements for Architecture Work

The business imperatives behind the Enterprise Architecture work drive the requirements and performance metrics for the architecture work. They should be sufficiently clear so that this phase may scope the business outcomes and resource requirements, and define the outline enterprise business information requirements and associated strategies of the Enterprise Architecture work to be done. For example, these may include:

- Business requirements
- Cultural aspirations
- Organization intents
- Strategic intent
- Forecast financial requirements

Significant elements of these need to be articulated so that the sponsor can identify all the key decision-makers and stakeholders involved in defining and establishing an Architecture Capability.

5.5.4 Principles

The Preliminary Phase defines the Architecture Principles that will form part of the constraints on any architecture work undertaken in the enterprise. The issues involved in this are explained in Part III, Chapter 20.

The definition of Architecture Principles is fundamental to the development of an Enterprise Architecture. Architecture work is informed by business principles as well as Architecture Principles. The Architecture Principles themselves are also normally based in part on business principles. Defining business principles normally lies outside the scope of the architecture function. However, depending on how such principles are defined and promulgated within the enterprise, it may be possible for the set of Architecture Principles to also restate, or cross-refer to a set of business principles, business goals, and strategic business drivers defined elsewhere within the enterprise. Within an architecture project, the architect will normally need to ensure that the definitions of these business principles, goals, and strategic drivers are current, and to clarify any areas of ambiguity.

The issue of Architecture Governance is closely linked to that of Architecture Principles. The body responsible for governance will also normally be responsible for approving the Architecture Principles, and for resolving architecture issues. The issues involved in governance are explained in Part VI, Chapter 44.

5.5.5 Management Frameworks

The TOGAF Architecture Development Method (ADM) is a generic method, intended to be used by enterprises in a wide variety of industry types and geographies. It is also designed for use with a wide variety of other Enterprise Architecture frameworks, if required (although it can be used perfectly well in its own right, without adaptation).

The TOGAF framework has to co-exist with and enhance the operational capabilities of other management frameworks that are present within any organization either formally or informally. In addition to these frameworks, most organizations have a method for the development of solutions, most of which have an IT component. The significance of systems is that they bring together the various domains (also known as People, Processes, and Material/Technology) to deliver a business capability.

The main frameworks suggested to be co-ordinated with the TOGAF framework are:

- **Business Capability Management** that determines what business capabilities are required to deliver business value including the definition of return on investment and the requisite control/performance measures
- **Project/Portfolio Management Methods** that determine how a company manages its change initiatives
- **Operations Management Methods** that describe how a company runs its day-to-day operations, including IT
- **Solution Development Methods** that formalize the way that business systems are delivered in accordance with the structures developed in the IT architecture

As illustrated in Figure 5-2, these frameworks are not discrete and there are significant overlaps between them and the Business Capability Management. The latter includes the delivery of performance measured business value.

The overall significance is that the Enterprise Architect applying the TOGAF framework cannot narrowly focus on the IT implementation, but must be aware of the impact that the architecture has on the entire enterprise.

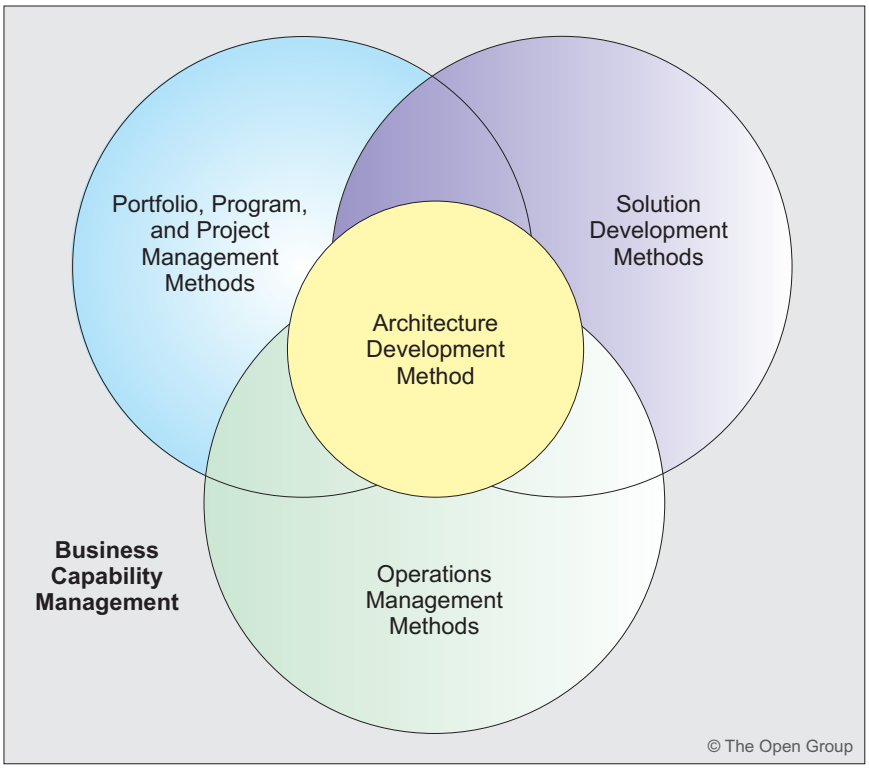

Figure 5-2 Management Frameworks to Co-ordinate with the TOGAF Framework

The Preliminary Phase therefore involves doing any necessary work to adapt the ADM to define an organization-specific framework, using either the TOGAF deliverables or the deliverables of another framework. The issues involved in this are discussed in Section 4.3.

5.5.6 Relating the Management Frameworks

Figure 5-3 illustrates a more detailed set of dependencies between the various frameworks and business planning activity that incorporates the enterprise's strategic plan and direction. The Enterprise Architecture can be used to provide a structure for all of the corporate initiatives, the Portfolio Management Framework can be used to deliver the components of the architecture, and the Operations Management Framework supports incorporation of these new components within the corporate infrastructure.

The business planners are present throughout the process and are in a position to support and enforce the architecture by retaining approval for resources at the various stages of planning and development.

The solution development methodology is used within the Portfolio Management Framework to plan, create, and deliver the architectural components specified in the project and portfolio charters. These deliverables include, but are not exclusively, IT; for example, a new building, a new set of skills, production equipment, hiring, marketing, and so on. Enterprise Architecture potentially provides the context for all enterprise activities.

The management frameworks are required to complement each other and work in close harmony for the good of the enterprise.

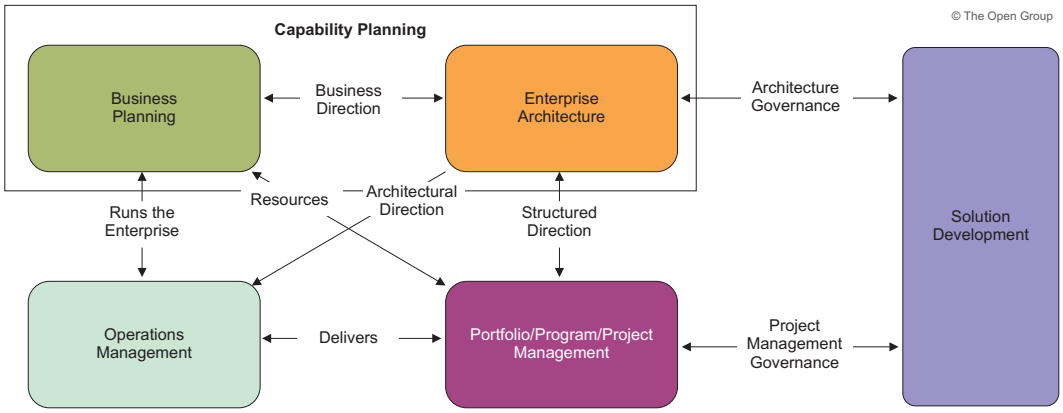

Figure 5-3 Interoperability and Relationships between Management Frameworks

Business planning at the strategy level provides the initial direction to Enterprise Architecture. Updates at the annual planning level provide a finer level of ongoing guidance. Capability-based planning is one of many popular techniques for business planning.

Enterprise Architecture structures the business planning into an integrated framework that regards the enterprise as a system or system of systems. This integrated approach will validate the business plan and can provide valuable feedback to the corporate planners. In some organizations, the Enterprise Architects have been moved to or work very closely with the strategic direction groups. The TOGAF approach delivers a framework for Enterprise Architecture.

Project/portfolio management is the delivery framework that receives the structured, detailed direction that enables them to plan and build what is required, knowing that each assigned deliverable will be in context (i.e., the piece of the puzzle that they deliver will fit into the corporate puzzle that is the Enterprise Architecture). Often this framework is based upon the Project Management Institute or UK Office of Government Commerce (PRINCE2) project management methodologies. Project architectures and detailed out-of-context design are often based upon systems design methodologies.

Operations management receives the deliverables and then integrates and sustains them within the corporate infrastructure. Often the IT service management services are based upon ISO/IEC 20000: 2011 or BS15000 (ITIL).

5.5.7 Planning for Enterprise Architecture/Business Change Maturity Evaluation

Capability Maturity Models (detailed in Part VI, Chapter 45) are useful ways of assessing the ability of an enterprise to exercise different capabilities.

Capability Maturity Models typically identify selected factors that are required to exercise a capability. An organization's ability to execute specific factors provides a measure of maturity and can be used to recommend a series of sequential steps to improve a capability. It is an assessment that gives executives an insight into pragmatically improving a capability.

A good Enterprise Architecture maturity model covers the characteristics necessary to develop and consume Enterprise Architecture. Organizations can determine their own factors and derive the appropriate maturity models, but it is recommended to take an existing model and customize it as required.

Several good models exist, including NASCIO, and the US Department of Commerce Architecture Capability Maturity Model.

The use of Capability Maturity Models is detailed in Part VI, Chapter 45.

Other examples include the US Federal Enterprise Architecture Maturity Model. Even though the models are originally from government, they are equally applicable to industry.

Chapter 6

Phase A: Architecture Vision

This chapter describes the initial phase of the Architecture Development Method (ADM). It includes information about defining the scope, identifying the stakeholders, creating the Architecture Vision, and obtaining approvals.

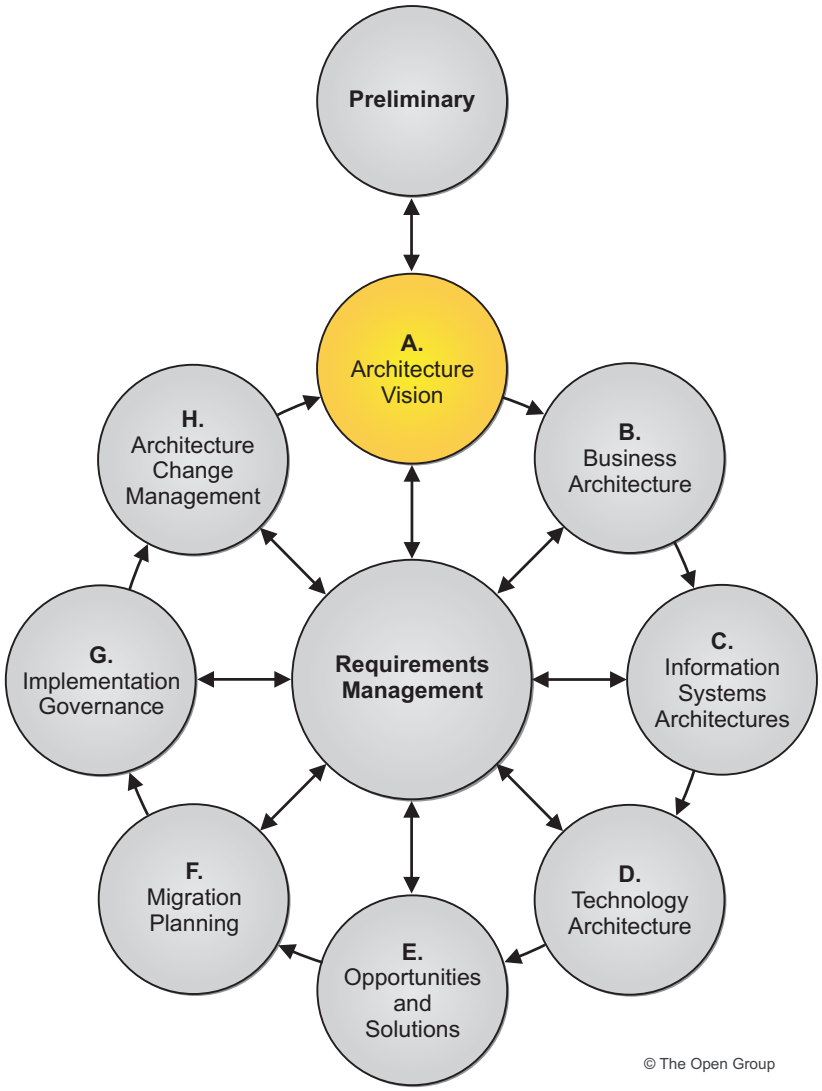

Figure 6-1 Phase A: Architecture Vision

6.1 Objectives

The objectives of Phase A are to:

- Develop a high-level aspirational vision of the capabilities and business value to be delivered as a result of the proposed Enterprise Architecture
- Obtain approval for a Statement of Architecture Work that defines a program of works to develop and deploy the architecture outlined in the Architecture Vision

6.2 Inputs

This section defines the inputs to Phase A.

6.2.1 Reference Materials External to the Enterprise

- Architecture reference materials (see Part IV, Section 32.2.5)

6.2.2 Non-Architectural Inputs

- Request for Architecture Work (see Part IV, Section 32.2.17)
- Business principles, business goals, and business drivers (see Part IV, Section 32.2.9)

6.2.3 Architectural Inputs

- Organizational Model for Enterprise Architecture (see Part IV, Section 32.2.16), including:
  - Scope of organizations impacted
  - Maturity assessment, gaps, and resolution approach
  - Roles and responsibilities for architecture team(s)
  - Constraints on architecture work
  - Re-use requirements
  - Budget requirements
  - Requests for change
  - Governance and support strategy
- Tailored Architecture Framework (see Part IV, Section 32.2.21), including:
  - Tailored architecture method
  - Tailored architecture content (deliverables and artifacts)
  - Architecture Principles (see Part IV, Section 32.2.4), including business principles, when pre-existing
  - Configured and deployed tools
- Populated Architecture Repository (see Part IV, Section 32.2.5) — existing architectural documentation (framework description, architectural descriptions, baseline descriptions, ABBs, etc.)

6.3 Steps

The level of detail addressed in Phase A will depend on the scope and goals of the Request for Architecture Work, or the subset of scope and goals associated with this iteration of architecture development.

The order of the steps in Phase A as well as the time at which they are formally started and completed should be adapted to the situation at hand in accordance with the established Architecture Governance.

The steps in Phase A are as follows:

- Establish the architecture project (see Section 6.3.1)
- Identify stakeholders, concerns, and business requirements (see Section 6.3.2)
- Confirm and elaborate business goals, business drivers, and constraints (see Section 6.3.3)
- Evaluate capabilities (see Section 6.3.4)
- Assess readiness for business transformation (see Section 6.3.5)
- Define scope (see Section 6.3.6)
- Confirm and elaborate Architecture Principles, including business principles (see Section 6.3.7)
- Develop Architecture Vision (see Section 6.3.8)
- Define the Target Architecture value propositions and KPIs (see Section 6.3.9)
- Identify the business transformation risks and mitigation activities (see Section 6.3.10)
- Develop Statement of Architecture Work; secure approval (see Section 6.3.11)

6.3.1 Establish the Architecture Project

Enterprise Architecture is a business capability; each cycle of the ADM should normally be handled as a project using the project management framework of the enterprise. In some cases, architecture projects will be stand-alone. In other cases, architectural activities will be a subset of the activities within a larger project. In either case, architecture activity should be planned and managed using accepted practices for the enterprise.

Conduct the necessary procedures to secure recognition of the project, the endorsement of corporate management, and the support and commitment of the necessary line management. Include references to other management frameworks in use within the enterprise, explaining how this project relates to those frameworks.

6.3.2 Identify Stakeholders, Concerns, and Business Requirements

Identify the key stakeholders and their concerns/objectives, and define the key business requirements to be addressed in the architecture engagement. Stakeholder engagement at this stage is intended to accomplish three objectives:

- To identify candidate vision components and requirements to be tested as the Architecture Vision is developed
- To identify candidate scope boundaries for the engagement to limit the extent of architectural investigation required

- To identify stakeholder concerns, issues, and cultural factors that will shape how the architecture is presented and communicated

The major product resulting from this step is a stakeholder map for the engagement, showing which stakeholders are involved with the engagement, their level of involvement, and their key concerns (see Part III, Section 21.3 and Section 21.4). The stakeholder map is used to support various outputs of the Architecture Vision phase, and to identify:

- The concerns and viewpoints that are relevant to this project; this is captured in the Architecture Vision (see Part IV, Section 32.2.8)
- The stakeholders that are involved with the project and as a result form the starting point for a Communications Plan (see Part IV, Section 32.2.12)
- The key roles and responsibilities within the project, which should be included within the Statement of Architecture Work (see Part VI, Section 32.2.20)

Another key task will be to consider which architecture views and viewpoints need to be developed to satisfy the various stakeholder requirements. As described in Part III, Chapter 21, understanding at this stage which stakeholders and which views need to be developed is important in setting the scope of the engagement.

During the Architecture Vision phase, new requirements generated for future architecture work within the scope of the selected requirements need to be documented within the Architecture Requirements Specification, and new requirements which are beyond the scope of the selected requirements must be input to the Requirements Repository for management through the Requirements Management process.

6.3.3 Confirm and Elaborate Business Goals, Business Drivers, and Constraints

Identify the business goals and strategic drivers of the organization.

If these have already been defined elsewhere within the enterprise, ensure that the existing definitions are current, and clarify any areas of ambiguity. Otherwise, go back to the originators of the Statement of Architecture Work and work with them to define these essential items and secure their endorsement by corporate management.

Define the constraints that must be dealt with, including enterprise-wide constraints and project-specific constraints (time, schedule, resources, etc.). The enterprise-wide constraints may be informed by the business and Architecture Principles developed in the Preliminary Phase or clarified as part of Phase A.

6.3.4 Evaluate Capabilities

It is valuable to understand a collection of capabilities within the enterprise. One part refers to the capability of the enterprise to develop and consume the architecture. The second part refers to the baseline and target capability level of the enterprise. Gaps identified in the Architecture Capability require iteration between Architecture Vision and Preliminary Phase to ensure that the Architecture Capability is suitable to address the scope of the architecture project (see Part III, Chapter 18).

A key step following from evaluation of business models, or artifacts that clarify priorities of a business strategy, is to identify the required business capabilities the enterprise must possess to act on the strategic priorities.

The detailed assessment of business capability gaps belongs in Phase B as a core aspect of the Business Architecture, where the architect can help the enterprise understand gaps throughout the business, of many types, that need to be addressed in later phases of the architecture.

In the Architecture Vision phase, however, the architect should consider the capability of the enterprise to develop the Enterprise Architecture itself, as required in the specific initiative or project underway. Gaps in the ability to progress through the ADM, whether deriving from skill shortages, information required, process weakness, or systems and tools, are a serious consideration in the vision of whether the architecture effort should continue. The architect can find guidance in Section 6.5 to gather existing business capability frameworks for the enterprise in this early assessment.

Gaps, or limitations, identified in the enterprise's capability to execute on change will inform the architect on the description of the Target Architecture and on the Implementation and Migration Plan (see Part IV, Section 32.2.14) created in Phase E and Phase F. This step seeks to understand the capabilities and desires of the enterprise at an appropriate level of abstraction (see Chapter 19). Consideration of the gap between the baseline and target capability of the enterprise is critical. Showing the baseline and target capabilities within the context of the overall enterprise can be supported by creating Value Chain diagrams that show the linkage of related capabilities. The results of the assessment are documented in a Capability Assessment (see (see Part IV, Section 32.2.10).

6.3.5 Assess Readiness for Business Transformation

A Business Transformation Readiness Assessment can be used to evaluate and quantify the organization's readiness to undergo a change. This assessment is based upon the determination and analysis/rating of a series of readiness factors, as described in Chapter 26.

The results of the readiness assessment should be added to the Capability Assessment (see Part IV, Section 32.2.10). These results are then used to shape the scope of the architecture, to identify activities required within the architecture project, and to identify risk areas to be addressed.

6.3.6 Define Scope

Define what is inside and what is outside the scope of the Baseline Architecture and Target Architecture efforts, understanding that the baseline and target need not be described at the same level of detail. In many cases, the baseline is described at a higher level of abstraction, so more time is available to specify the target in sufficient detail. The issues involved in this are discussed in Section 4.5. In particular, define:

- The breadth of coverage of the enterprise
- The level of detail required
- The partitioning characteristics of the architecture (see Part V, Chapter 36 for more details)
- The specific architecture domains to be covered (business, data, application, technology)
- The extent of the time period aimed at, plus the number and extent of any intermediate time period
- The architectural assets to be leveraged, or considered for use, from the organization's Enterprise Continuum:

— Assets created in previous iterations of the ADM cycle within the enterprise

— Assets available elsewhere in the industry (other frameworks, systems models, vertical industry models, etc.)

6.3.7 Confirm and Elaborate Architecture Principles, including Business Principles

Review the principles under which the architecture is to be developed. Architecture Principles are normally based on the principles developed as part of the Preliminary Phase. They are explained, and an example set given, in Part III, Chapter 20. Ensure that the existing definitions are current, and clarify any areas of ambiguity. Otherwise, go back to the body responsible for Architecture Governance and work with them to define these essential items for the first time and secure their endorsement by corporate management.

6.3.8 Develop Architecture Vision

An understanding of the required artifacts will enable the stakeholders to start to scope out their decision-making which will guide subsequent phases. These decisions need to be reflected in the stakeholder map.

Policy development and strategic decisions need to be captured in this phase to enable the subsequent work to be quantified; for example, rationalization decisions and metrics, revenue generation, and targets which meet the business strategy. There are also other areas which need to be addressed; for example, Digital Transformation and IT strategy where decisions on the Architecture Vision will provide leadership and direction for the organization in subsequent phases.

For the Architecture Vision it is recommended that first an overall architecture be decided upon showing how all of the various architecture domain deliverables will fit together (based upon the selected course of action).

Based on the stakeholder concerns, business capability requirements, scope, constraints, and principles, create a high-level view of the Baseline and Target Architectures. The Architecture Vision typically covers the breadth of scope identified for the project, at a high level. Informal techniques are often employed. A common practice is to draw a simple solution concept diagram that illustrates concisely the major components of the solution and how the solution will result in benefit for the enterprise.

Business scenarios are an appropriate and useful technique to discover and document business requirements, and to articulate an Architecture Vision that responds to those requirements. Business scenarios may also be used at more detailed levels of the architecture work (e.g., in Phase B) and are described in the TOGAF® Series Guide: Business Scenarios.

This step generates the first, very high-level definitions of the baseline and target environments, from a business, information systems, and technology perspective, as described in Section 6.4.

These initial versions of the architecture should be stored in the Architecture Repository, organized according to the standards and guidelines established in the architecture framework.

6.3.9 Define the Target Architecture Value Propositions and KPIs

- Develop the business case for the architectures and changes required
- Produce the value proposition for each of the stakeholder groupings
- Assess and define the procurement requirements
- Review and agree the value propositions with the sponsors and stakeholders concerned
- Define the performance metrics and measures to be built into the Enterprise Architecture to meet the business needs
- Assess the business risk (see Part III, Chapter 27)

The outputs from this activity should be incorporated within the Statement of Architecture Work to allow performance to be tracked accordingly.

6.3.10 Identify the Business Transformation Risks and Mitigation Activities

Identify the risks associated with the Architecture Vision and assess the initial level of risk (e.g., catastrophic, critical, marginal, or negligible) and the potential frequency associated with it. Assign a mitigation strategy for each risk. A risk management framework is described in Part III, Chapter 27.

There are two levels of risk that should be considered, namely:

- **Initial Level of Risk**: risk categorization prior to determining and implementing mitigating actions
- **Residual Level of Risk**: risk categorization after implementation of mitigating actions (if any)

Risk mitigation activities should be considered for inclusion within the Statement of Architecture Work.

6.3.11 Develop Statement of Architecture Work; Secure Approval

Assess the work products that are required to be produced (and by when) against the set of business performance requirements. This will involve ensuring that:

- Performance metrics are built into the work products
- Specific performance-related work products are available

Then, activities will include:

- Identify new work products that will need to be changed
- Provide direction on which existing work products, including building blocks, will need to be changed and ensure that all activities and dependencies on these are co-ordinated
- Identify the impact of change on other work products and dependence on their activities
- Based on the purpose, focus, scope, and constraints, determine which architecture domains should be developed, to what level of detail, and which architecture views should be built
- Assess the resource requirements and availability to perform the work in the timescale required; this will include adhering to the organization's planning methods and work products to produce the plans for performing a cycle of the ADM

- Estimate the resources needed, develop a roadmap and schedule for the proposed development, and document all these in the Statement of Architecture Work
- Define the performance metrics to be met during this cycle of the ADM by the Enterprise Architecture team
- Develop the specific Enterprise Architecture Communications Plan and show where, how, and when the Enterprise Architects will communicate with the stakeholders, including affinity groupings and communities, about the progress of the Enterprise Architecture developments
- Review and agree the plans with the sponsors, and secure formal approval of the Statement of Architecture Work under the appropriate governance procedures
- Gain sponsor's sign-off to proceed

6.4 Outputs

The outputs of Phase A may include, but are not restricted to:

- Approved Statement of Architecture Work (see Part IV, Section 32.2.20), including in particular:
  - Architecture project description and scope
  - Overview of Architecture Vision
  - Architecture project plan and schedule
- Refined statements of business principles, business goals, and business drivers (see Part IV, Section 32.2.9)
- Architecture Principles (see Part IV, Chapter 20)
- Capability Assessment (see Part IV, Section 32.2.10)
- Tailored Architecture Framework (see Part IV, Section 32.2.21) (for the engagement), including:
  - Tailored architecture method
  - Tailored architecture content (deliverables and artifacts)
  - Configured and deployed tools
- Architecture Vision (see Part IV, Section 32.2.8), including:
  - Problem description
  - Objective of the Statement of Architecture Work
  - Summary views
  - Business Scenario (optional)
  - Refined key high-level stakeholder requirements
- Draft Architecture Definition Document, including (when in scope):
  - Baseline Business Architecture, Version 0.1
  - Baseline Technology Architecture, Version 0.1

  - Baseline Data Architecture, Version 0.1
  - Baseline Application Architecture, Version 0.1
  - Target Business Architecture, Version 0.1
  - Target Technology Architecture, Version 0.1
  - Target Data Architecture, Version 0.1
  - Target Application Architecture, Version 0.1
- Communications Plan (see Part IV, Section 32.2.12)
- Additional content populating the Architecture Repository (see Part IV, Section 32.2.5)

Note: Multiple business scenarios may be used to generate a single Architecture Vision.

The outputs may include some or all of the following:

- Matrices:
  - Stakeholder Map matrix
- Diagrams:
  - Business Model diagram
  - Business Capability Map
  - Value Stream Map
  - Value Chain diagram
  - Solution Concept diagram

6.5 Approach

6.5.1 General

Phase A starts with receipt of a Request for Architecture Work from the sponsoring organization to the architecture organization.

The issues involved in ensuring proper recognition and endorsement from corporate management, and the support and commitment of line management, are discussed in Part VI, Section 44.1.4.

Phase A also defines what is in and what is outside the scope of the architecture effort and the constraints that must be dealt with. Scoping decisions need to be made on the basis of a practical assessment of resource and competence availability, and the value that can realistically be expected to accrue to the enterprise from the chosen scope of architecture work. The issues involved in this are discussed in Section 4.5. Scoping issues addressed in the Architecture Vision phase will be restricted to the specific objectives for this ADM cycle and will be constrained within the overall scope definition for architecture activity as established within the Preliminary Phase and embodied within the architecture framework.

In situations where the architecture framework in place is not appropriate to achieve the desired Architecture Vision, revisit the Preliminary Phase and extend the overall architecture framework for the enterprise.

The constraints will normally be informed by the business principles and Architecture Principles, developed as part of the Preliminary Phase (see Chapter 5).

Normally, the business principles, business goals, and strategic drivers of the organization are already defined elsewhere in the enterprise. If so, the activity in Phase A is involved with ensuring that existing definitions are current, and clarifying any areas of ambiguity. Otherwise, it involves defining these essential items for the first time.

Similarly, the Architecture Principles that form part of the constraints on architecture work will normally have been defined in the Preliminary Phase (see Chapter 5). The activity in Phase A is concerned with ensuring that the existing principle definitions are current, and clarifying any areas of ambiguity. Otherwise, it entails defining the Architecture Principles for the first time, as explained in Part III, Chapter 20.

6.5.2 Creating the Architecture Vision

The Architecture Vision provides the sponsor with a key tool to sell the benefits of the proposed capability to stakeholders and decision-makers within the enterprise. Architecture Vision describes how the new capability will meet the business goals and strategic objectives and address the stakeholder concerns when implemented.

Integral to the Architecture Vision is an understanding of emerging technologies and their potential impact on industries and enterprises, without which many business opportunities may be missed.

Clarifying and agreeing the purpose of the architecture effort is one of the key parts of this activity, and the purpose needs to be clearly reflected in the vision that is created. Architecture projects are often undertaken with a specific purpose in mind — a specific set of business drivers that represent the return on investment for the stakeholders in the architecture development. Clarifying that purpose, and demonstrating how it will be achieved by the proposed architecture development, is the whole point of the Architecture Vision.

Normally, key elements of the Architecture Vision — such as the enterprise mission, vision, strategy, and goals — have been documented as part of some wider business strategy or enterprise planning activity that has its own lifecycle within the enterprise. In such cases, the activity in Phase A is concerned with verifying and understanding the documented business strategy and goals, and possibly bridging between the enterprise strategy and goals on the one hand, and the strategy and goals implicit within the current architecture reality.

Business models are key strategy artifacts that can provide such a perspective, by showing how the organization intends to deliver value to its customers and stakeholders. Section 6.3.4 introduces the application of business models as a step in developing the Architecture Vision.

In other cases, little or no Business Architecture work may have been done to date. In such cases, there will be a need for the architecture team to research, verify, and gain buy-in to the key business objectives and processes that the architecture is to support. This may be done as a free-standing exercise, either preceding architecture development, or as part of the ADM initiation phase (Preliminary Phase).

This exercise should examine and search for existing materials on fundamental Business Architecture concepts such as:

- **Business Capabilities**, which represent a particular ability or capacity that a business may possess or exchange to achieve a specific purpose or outcome

  In this phase, the architect should determine whether a framework exists in the organization to represent business capabilities. If one does not exist, the architect should consider whether developing a framework is within the scope of the project. For an introduction to business capabilities, see The Open Group Guide to Business Capabilities.

- **Value Streams**, which represent an end-to-end collection of value-adding activities that create an overall result for a customer, stakeholder, or end user

  For an introduction to value streams, see the TOGAF® Series Guide: Value Streams.

- **Organization Maps**, which depict the relationships between the primary entities that make up the enterprise, its partners, and stakeholders

  As traditional organizational charts often lack the necessary detail to reflect the full scope of the enterprise's activities, the architect can help identify and understand the complex web of relationships between business entities as well as where business capabilities are used and connection to value stream stages. These are refined and extended in subsequent phases.

In addition, the Architecture Vision explores other domains which are appropriate for the Enterprise Architecture in hand. These domains may include elements of the basic domains, yet serve an additional purpose for the stakeholders. Example domains may include:

- Information
- Security
- Digital
- Network Management
- Knowledge
- Industry-specific
- Services
- Partnership
- Cybersecurity

These domains may be free-standing or linked with other domains to provide enterprise-wide views of the organization vision and structure.

The Architecture Vision phase includes the conduct of a business assessment (using, for example, business scenarios) where critical factors are documented and various courses of action are assessed. High-level advantages and disadvantages, including risks and opportunities, are documented and the best course of action selected to serve as the basis for the Architecture Vision.

The Architecture Vision provides a first-cut, high-level description of the Baseline and Target Architectures, covering the business, data, application, and technology domains. These outline descriptions are developed in subsequent phases.

Once an Architecture Vision is defined and documented in the Statement of Architecture Work, it is critical to use it to build a consensus, as described in Part VI, Section 44.1.4. Without this consensus it is very unlikely that the final architecture will be accepted by the organization as a whole. The consensus is represented by the sponsoring organization signing the Statement of Architecture Work.

Chapter 7

Phase B: Business Architecture

This chapter describes the development of a Business Architecture to support an agreed Architecture Vision.

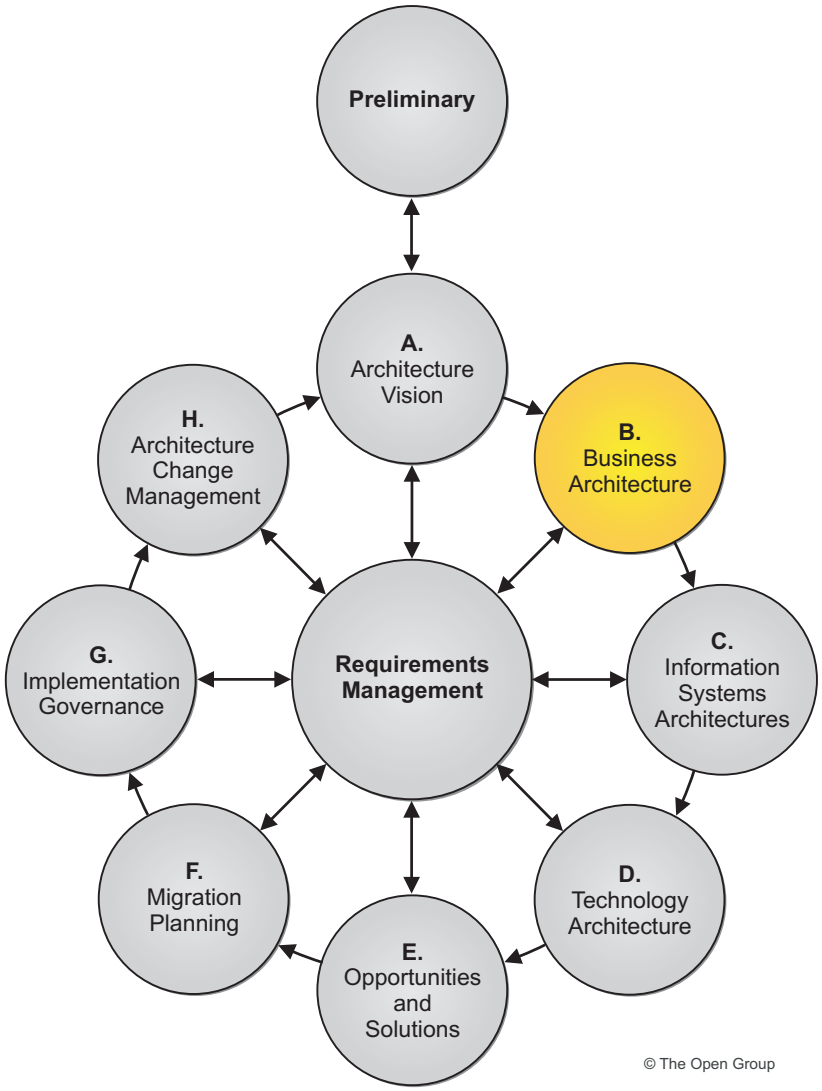

Figure 7-1 Phase B: Business Architecture

7.1 Objectives

The objectives of Phase B are to:

- Develop the Target Business Architecture that describes how the enterprise needs to operate to achieve the business goals, and respond to the strategic drivers set out in the Architecture Vision, in a way that addresses the Statement of Architecture Work and stakeholder concerns
- Identify candidate Architecture Roadmap components based upon gaps between the Baseline and Target Business Architectures

7.2 Inputs

This section defines the inputs to Phase B.

7.2.1 Reference Materials External to the Enterprise

- Architecture reference materials (see Part IV, Section 32.2.5)

7.2.2 Non-Architectural Inputs

- Request for Architecture Work (see Part IV, Section 32.2.17)
- Business principles, business goals, and business drivers (see Part IV, Section 32.2.9)
- Capability Assessment (see Part IV, Section 32.2.10)
- Communications Plan (see Part IV, Section 32.2.12)

7.2.3 Architectural Inputs

- Organizational Model for Enterprise Architecture (see Part IV, Section 32.2.16), including:
  - 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
- Tailored Architecture Framework (see Part IV, Section 32.2.21), including:
  - Tailored architecture method
  - Tailored architecture content (deliverables and artifacts)
  - Configured and deployed tools
- Approved Statement of Architecture Work (see Part IV, Section 32.2.20)

- Architecture Principles (see Part IV, Section 32.2.4), including business principles, when pre-existing
- Enterprise Continuum (see Part V, Chapter 35)
- Architecture Repository (see Part IV, Section 32.2.5), including:
  - Re-usable building blocks
  - Publicly available reference models
  - Organization-specific reference models
  - Organization standards
- Architecture Vision (see Part IV, Section 32.2.8), including:
  - Problem description
  - Objective of the Statement of Architecture Work
  - Summary views
  - Business Scenario (optional)
  - Refined key high-level stakeholder requirements
- Draft Architecture Definition Document, including (when in scope):
  - Baseline Business Architecture, Version 0.1
  - Baseline Technology Architecture, Version 0.1
  - Baseline Data Architecture, Version 0.1
  - Baseline Application Architecture, Version 0.1
  - Target Business Architecture, Version 0.1
  - Target Technology Architecture, Version 0.1
  - Target Data Architecture, Version 0.1
  - Target Application Architecture, Version 0.1

7.3 Steps

The level of detail addressed in Phase B will depend on the scope and goals of the overall architecture effort.

New models characterizing the needs of the business will need to be defined in detail during Phase B. Existing business artifacts to be carried over and supported in the target environment may already have been adequately defined in previous architectural work; but, if not, they too will need to be defined in Phase B.

The order of the steps in Phase B as well as the time at which they are formally started and completed should be adapted to the situation at hand, in accordance with the established Architecture Governance. In particular, determine whether in this situation it is appropriate to conduct Baseline or Target Architecture development first, as described in Part III, Chapter 18.

All activities that have been initiated in these steps should be closed during the Finalize the Business Architecture step (see Section 7.3.8). The documentation generated from these steps must be formally published in the Create the Architecture Definition Document step (see Section 7.3.9).

The steps in Phase B are as follows:

- Select reference models, viewpoints, and tools (see Section 7.3.1)
- Develop Baseline Business Architecture Description (see Section 7.3.2)
- Develop Target Business Architecture Description (see Section 7.3.3)
- Perform gap analysis (see Section 7.3.4)
- Define candidate roadmap components (see Section 7.3.5)
- Resolve impacts across the Architecture Landscape (see Section 7.3.6)
- Conduct formal stakeholder review (see Section 7.3.7)
- Finalize the Business Architecture (see Section 7.3.8)
- Create the Architecture Definition Document (see Section 7.3.9)

7.3.1 Select Reference Models, Viewpoints, and Tools

Select relevant Business Architecture resources (reference models, patterns, etc.) from the Architecture Repository, on the basis of the business drivers, and the stakeholders and concerns.

Select relevant Business Architecture viewpoints (e.g., operations, management, financial); i.e., those that will enable the architect to demonstrate how the stakeholder concerns are being addressed in the Business Architecture.

Identify appropriate tools and techniques to be used for capture, modeling, and analysis, in association with the selected viewpoints. Depending on the degree of sophistication warranted, these may comprise simple documents or spreadsheets, or more sophisticated modeling tools and techniques, such as activity models, business process models, use-case models, etc.

7.3.1.1 Determine Overall Modeling Process

Business modeling and strategy assessments are effective techniques for framing the target state of an organization's Business Architecture (see Section 6.3.4). The output from that activity is then used to articulate the business capabilities, organizational structure, and value streams required to close gaps between the current and target state. As addressed in Section 6.5, the frameworks for these maps may already exist and should be leveraged, now using them to characterize gaps and better mapping of business value to achieve the Target Business Architecture.

For each viewpoint, select the models needed to support the specific view required, using the selected tool or method.

Ensure that all stakeholder concerns are covered. If they are not, create new models to address concerns not covered, or augment existing models (see Section 7.5.6). Business scenarios are a useful technique to discover and document business requirements, and may be used iteratively, at different levels of detail in the hierarchical decomposition of the Business Architecture. Business scenarios are described in the TOGAF® Series Guide: Business Scenarios.

The techniques described in Section 7.5 can be utilized to progressively decompose a business:

- **Business Capability Mapping**: identifies, categorizes, and decomposes the business capabilities required for the business to have the ability to deliver value to one or more stakeholders
- **Organization Mapping**: a representation of the organizational structure of the business (including third-party domains), depicting business units, the decomposition of those units into lower-level functions, and organizational relationships (unit-to-unit and mapping to business capabilities, locations, and other attributes)
- **Value Stream Mapping**: the breakdown of activities that an organization performs to create the value being exchanged with stakeholders

  Value stream maps illustrate how an organization delivers value and are in the context of a specific set of stakeholders, and leverage business capabilities in order to create stakeholder value and align to other aspects of the Target Business Architecture.
- **Structured Analysis**: identifies the key business functions within the scope of the architecture, and maps those functions onto the organizational units within the business
- **Use-case Analysis**: the breakdown of business-level functions across actors and organizations allows the actors in a function to be identified and permits a breakdown into services supporting/delivering that functional capability
- **Process Modeling**: the breakdown of a function or business service through process modeling allows the elements of the process to be identified, and permits the identification of lower-level business services or functions

The level and rigor of decomposition needed varies from enterprise to enterprise, as well as within an enterprise, and the architect should consider the enterprise's goals, objectives, scope, and purpose of the Enterprise Architecture effort to determine the level of decomposition.

7.3.1.2 Identify Required Service Granularity Level, Boundaries, and Contracts

The TOGAF content framework differentiates between the functions of a business and the services of a business. Business services are specific functions that have explicit, defined boundaries that are explicitly governed. In order to allow the architect flexibility to define business services at a level of granularity that is appropriate for and manageable by the business, the functions are split as follows: micro-level functions will have explicit, defined boundaries, but may not be explicitly governed. Likewise, macro business functions may be explicitly governed, but may not have explicit, defined boundaries.

The Business Architecture phase therefore needs to identify which components of the architecture are functions and which are services. Services are distinguished from functions through the explicit definition of a service contract. When Baseline Architectures are being developed, it may be the case that explicit contracts do not exist and it would therefore be at the discretion of the architect to determine whether there is merit in developing such contracts before examining any Target Architectures.

A service contract covers the business/functional interface and also the technology/data interface. The Business Architecture will define the service contract at the business/functional level, which will be expanded on in the Application and Technology Architecture phases.

The granularity of business services should be determined according to the business drivers, goals, objectives, and measures for this area of the business. Finer-grained services permit closer management and measurement (and can be combined to create coarser-grained services), but require greater effort to govern. Guidelines for identification of services and definition of their

contracts can be found in the TOGAF® Series Guide: Using the TOGAF® Framework to Define and Govern Service-Oriented Architectures.

7.3.1.3 *Identify Required Catalogs of Business Building Blocks*

Catalogs capture inventories of the core assets of the business. Catalogs are hierarchical in nature and capture the decomposition of a building block and also decompositions across related building blocks (e.g., organization/actor).

Catalogs form the raw material for development of matrices and views and also act as a key resource for managing the business and IT capability.

The following catalogs should be considered for development within a Business Architecture:

- Value Stream catalog
- Business Capabilities catalog
- Value Stream Stages catalog
- Organization/Actor catalog
- Driver/Goal/Objective catalog
- Role catalog
- Business Service/Function catalog
- Location catalog
- Process/Event/Control/Product catalog
- Contract/Measure catalog

The structure of catalogs is based on the attributes of metamodel entities, as defined in Part IV, Chapter 30.

7.3.1.4 *Identify Required Matrices*

Matrices show the core relationships between related model entities.

Matrices form the raw material for development of views and also act as a key resource for impact assessment, carried out as a part of gap analysis.

The following matrices should be considered for development within a Business Architecture:

- Value Stream/Capability matrix (displays the capabilities required to support each stage of a value stream)
- Strategy/Capability matrix (displays the capabilities required to support specific strategy statements)
- Capability/Organization matrix (displays organization elements that implement each capability)
- Business Interaction matrix (showing dependency and communication between organizations and actors)
- Actor/Role matrix (showing the roles undertaken by each actor)

The structure of matrices is based on the attributes of metamodel entities, as defined in Part IV, Chapter 30.

7.3.1.5 Identify Required Diagrams

Diagrams present the Business Architecture information from a set of different perspectives (viewpoints) according to the requirements of the stakeholders.

The following diagrams should be considered for development within a Business Architecture:

- Business Model diagram
- Business Capability Map
- Value Stream Map
- Organization Map
- Business Footprint diagram
- Business Service/Information diagram
- Functional Decomposition diagram
- Goal/Objective/Service diagram
- Business Use-Case diagram
- Organization Decomposition diagram
- Process Flow diagram
- Event diagram

The structure of diagrams is based on the attributes of metamodel entities, as defined in Part IV, Chapter 30.

7.3.1.6 Identify Types of Requirement to be Collected

Once the Business Architecture catalogs, matrices, and diagrams have been developed, architecture modeling is completed by formalizing the business-focused requirements for implementing the Target Architecture.

These requirements may:

- Relate to the business domain
- Provide requirements input into the Data, Application, and Technology Architectures
- Provide detailed guidance to be reflected during design and implementation to ensure that the solution addresses the original architecture requirements

Within this step, the architect should identify requirements that should be met by the architecture (see Section 16.5.2).

In many cases, the Architecture Definition will not be intended to give detailed or comprehensive requirements for a solution (as these can be better addressed through general requirements management discipline). The expected scope of requirements content should be established during the Architecture Vision phase and documented in the approved Statement of Architecture Work.

Any requirement, or change in requirement, that is outside of the scope defined in the Statement of Architecture Work must be submitted to the Requirements Repository for management through the governed Requirements Management process.

7.3.2 Develop Baseline Business Architecture Description

Develop a Baseline Description of the existing Business Architecture, to the extent necessary to support the Target Business Architecture. The scope and level of detail to be defined will depend on the extent to which existing business elements are likely to be carried over into the Target Business Architecture, and on whether Architecture Descriptions exist, as described in Section 7.5. To the extent possible, identify the relevant Business Architecture building blocks, drawing on the Architecture Repository (see Part V, Chapter 37).

Where new architecture models need to be developed to satisfy stakeholder concerns, use the models identified within Step 1 as a guideline for creating new architecture content to describe the Baseline Architecture.

7.3.3 Develop Target Business Architecture Description

Develop a Target Description for the Business Architecture, to the extent necessary to support the Architecture Vision. The scope and level of detail to be defined will depend on the relevance of the business elements to attaining the Target Architecture Vision, and on whether architectural descriptions exist. To the extent possible, identify the relevant Business Architecture building blocks, drawing on the Architecture Repository (see Part V, Chapter 37).

Where new architecture models need to be developed to satisfy stakeholder concerns, use the models identified within Step 1 as a guideline for creating new architecture content to describe the Target Architecture.

7.3.4 Perform Gap Analysis

Verify the architecture models for internal consistency and accuracy:

- Perform trade-off analysis to resolve conflicts (if any) among the different views
- Validate that the models support the principles, objectives, and constraints
- Note changes to the viewpoint represented in the selected models from the Architecture Repository, and document
- Test architecture models for completeness against requirements

Identify gaps between the baseline and target, using the gap analysis technique as described in Part III, Chapter 23.

7.3.5 Define Candidate Roadmap Components

Following the creation of a Baseline Architecture, Target Architecture, and gap analysis results, a business roadmap is required to prioritize activities over the coming phases.

This initial Business Architecture Roadmap will be used as raw material to support more detailed definition of a consolidated, cross-discipline roadmap within the Opportunities & Solutions phase.

7.3.6 Resolve Impacts Across the Architecture Landscape

Once the Business Architecture is finalized, it is necessary to understand any wider impacts or implications.

At this stage, other architecture artifacts in the Architecture Landscape should be examined to identify:

- Does this Business Architecture create an impact on any pre-existing architectures?
- Have recent changes been made that impact on the Business Architecture?
- Are there any opportunities to leverage work from this Business Architecture in other areas of the organization?
- Does this Business Architecture impact other projects (including those planned as well as those currently in progress)?
- Will this Business Architecture be impacted by other projects (including those planned as well as those currently in progress)?

7.3.7 Conduct Formal Stakeholder Review

Check the original motivation for the architecture project and the Statement of Architecture Work against the proposed Business Architecture, asking if it is fit for the purpose of supporting subsequent work in the other architecture domains. Refine the proposed Business Architecture only if necessary.

7.3.8 Finalize the Business Architecture

- Select standards for each of the building blocks, re-using as much as possible from the reference models selected from the Architecture Repository
- Fully document each building block
- Conduct a final cross-check of overall architecture against business goals; document the rationale for building block decisions in the architecture document
- Document the final requirements traceability report
- Document the final mapping of the architecture within the Architecture Repository; from the selected building blocks, identify those that might be re-used (working practices, roles, business relationships, job descriptions, etc.), and publish via the Architecture Repository
- Finalize all the work products, such as gap analysis results

7.3.9 Create the Architecture Definition Document

- Document the rationale for building block decisions in the Architecture Definition Document
- Prepare the business sections of the Architecture Definition Document, comprising some or all of:
  - A business footprint (a high-level description of the people and locations involved with key business functions)

- A detailed description of business functions and their information needs
- A management footprint (showing span of control and accountability)
- Standards, rules, and guidelines showing working practices, legislation, financial measures, etc.
- A skills matrix and set of job descriptions

If appropriate, use reports and/or graphics generated by modeling tools to demonstrate key views of the architecture. Route the document for review by relevant stakeholders, and incorporate feedback.

7.4 Outputs

The outputs of Phase B may include, but are not restricted to:

- Refined and updated versions of the Architecture Vision phase deliverables, where applicable, including:
  - Statement of Architecture Work (see Part IV, Section 32.2.20), updated if necessary
  - Validated business principles, business goals, and business drivers (see Part IV, Section 32.2.9), updated if necessary
  - Architecture Principles (see Part IV, Section 32.2.4)
- Draft Architecture Definition Document (see Part IV, Section 32.2.3), including:
  - Baseline Business Architecture, Version 1.0 (detailed), if appropriate
  - Target Business Architecture, Version 1.0 (detailed), including:
    - Organization structure — identifying business locations and relating them to organizational units
    - Business goals and objectives — for the enterprise and each organizational unit
    - Business functions — a detailed, recursive step involving successive decomposition of major functional areas into sub-functions
    - Business services — the services that the enterprise and each enterprise unit provides to its customers, both internally and externally
    - Business processes, including measures and deliverables
    - Business roles, including development and modification of skills requirements
    - Business data model
    - Correlation of organization and functions — relate business functions to organizational units in the form of a matrix report
  - Views corresponding to the selected viewpoints addressing key stakeholder concerns
- Draft Architecture Requirements Specification (see Part IV, Section 32.2.6, on page 354), including such Business Architecture requirements as:
  - Gap analysis results
  - Technical requirements — identifying, categorizing, and prioritizing the implications for work in the remaining architecture domains; for example, by a dependency/priority matrix (for example, guiding trade-off between speed of transaction processing and security); list the specific models that are expected to be

produced (for example, expressed as primitives of the Zachman Framework)

- Updated business requirements

- Business Architecture components of an Architecture Roadmap (see Part IV, Section 32.2.7)

The outputs may include some or all of the following:

- Catalogs:
  - Value Stream catalog
  - Business Capabilities catalog
  - Value Stream Stages catalog
  - Organization/Actor catalog
  - Driver/Goal/Objective catalog
  - Role catalog
  - Business Service/Function catalog
  - Location catalog
  - Process/Event/Control/Product catalog
  - Contract/Measure catalog
- Matrices:
  - Value Stream/Capability matrix
  - Strategy/Capability matrix
  - Capability/Organization matrix
  - Business Interaction matrix
  - Actor/Role matrix
- Diagrams:
  - Business Model diagram
  - Business Capability Map
  - Value Stream Map
  - Organization Map
  - Business Footprint diagram
  - Business Service/Information diagram
  - Functional Decomposition diagram
  - Product Lifecycle diagram
  - Goal/Objective/Service diagram
  - Business Use-Case diagram
  - Organization Decomposition diagram

— Process Flow diagram

— Event diagram

7.5 Approach

Business Architecture is a representation of holistic, multi-dimensional business views of: capabilities, end-to-end value delivery, information, and organizational structure; and the relationships among these business views and strategies, products, policies, initiatives, and stakeholders.

Business Architecture relates business elements to business goals and elements of other domains.

7.5.1 General

A knowledge of the Business Architecture is a prerequisite for architecture work in any other domain (Data, Application, Technology), and is therefore the first architecture activity that needs to be undertaken, if not catered for already in other organizational processes (enterprise planning, strategic business planning, business process re-engineering, etc.).

In practical terms, the Business Architecture is also often necessary as a means of demonstrating the business value of subsequent architecture work to key stakeholders, and the return on investment to those stakeholders from supporting and participating in the subsequent work.

The scope of work in Phase B is primarily determined by the Architecture Vision as set out in Phase A. The business strategy defines the goals and drivers and metrics for success, but not necessarily how to get there. That is the role of the Business Architecture, defined in detail in Phase B.

This will depend to a large extent on the enterprise environment. In some cases, key elements of the Business Architecture may be done in other activities; for example, the enterprise mission, vision, strategy, and goals may be documented as part of some wider business strategy or enterprise planning activity that has its own lifecycle within the enterprise.

In such cases, there may be a need to verify and update the currently documented business strategy and plans, and/or to bridge between high-level business drivers, business strategy, and goals on the one hand, and the specific business requirements that are relevant to this architecture development effort. The business strategy typically defines what to achieve — the goals and drivers, and the metrics for success — but not how to get there. That is the role of the Business Architecture.

In other cases, little or no Business Architecture work may have been done to date. In such cases, there will be a need for the architecture team to research, verify, and gain buy-in to the key business objectives and processes that the architecture is to support. This may be done as a free-standing exercise, either preceding architecture development, or as part of Phase A.

In both of these cases, the business scenarios technique (see the TOGAF® Series Guide: Business Scenarios), or any other method that illuminates the key business requirements and indicates the implied technical requirements for IT architecture, may be used.

A key objective is to re-use existing material as much as possible. In architecturally more mature environments, there will be existing Architecture Definitions, which (hopefully) will have been maintained since the last architecture development cycle. Where Architecture Descriptions exist, these can be used as a starting point, and verified and updated if necessary; see Part V, Section 35.4.1.

Gather and analyze only that information that allows informed decisions to be made relevant to the scope of this architecture effort. If this effort is focused on the definition of (possibly new) business processes, then Phase B will necessarily involve a lot of detailed work. If the focus is more on the Target Architectures in other domains (data/information, application systems, infrastructure) to support an essentially existing Business Architecture, then it is important to build a complete picture in Phase B without going into unnecessary detail.

7.5.2 Developing the Baseline Description

If an enterprise has existing Architecture Descriptions, they should be used as the basis for the Baseline Description. This input may have been used already in Phase A in developing the Architecture Vision, such as the business capability map or a core set of value streams as introduced in Section 6.5.2, and may be sufficient in itself for this baseline.

The reasons to update these materials include having a missing business capability, a new value stream, or changed organizational unit that has not previously been assessed within the scope of the Enterprise Architecture project. Section 7.5.3 to Section 7.5.5 address the use of core Business Architecture methods to model the Business Architecture driven by the strategy scope from Phase A. Note that putting these methods into action to drive a focus and target state for later architecture work does not mean the fundamental frameworks from Phase A, such as a common enterprise business capability map, necessarily change but rather that they are applied in a manner driven by the scope and needs of the specific Enterprise Architecture project.

If no Architecture Descriptions exist, information should be gathered and Business Architecture models developed.

Whatever the scope of the specific project, it is important to determine whether it is the fundamental view of the business that is changing or the usage of those views to determine scope, priorities, and relationships for the specific project in relation to the rest of the enterprise.

7.5.3 Applying Business Capabilities

The business capability map found or developed in the Architecture Vision phase provides a self-contained view of the business that is independent of the current organizational structure, business processes, information systems and applications, and the rest of the product or service portfolio. Those business capabilities should be mapped back to the organizational units, value streams, information systems, and strategic plans within the scope of the Enterprise Architecture project. This relationship mapping provides greater insight into the alignment and optimization of each of those domains (see Relationship Mapping in The Open Group Guide to Business Capabilities).

Another common analysis technique involves heat mapping, which can be used to show a range of different perspectives on the same set of core business capabilities. These include maturity, effectiveness, performance, and the value or cost of each capability to the business. Different attributes determine the colors of each capability on the business capability map (see Heat Mapping in The Open Group Guide to Business Capabilities).

For example, a business capability maturity heat map shows the desired maturity as green for a specific capability, one level down as yellow, and two or more levels down as red. Other colors may indicate status, such as purple denoting a capability that does not exist yet in the company but is desired, or perhaps as a capability that is over-funded and has more resources than necessary. This gap analysis is directly tied to the Enterprise Architecture project underway; a gap is only relevant in the context of the business need and provides focus for more mapping in this phase or priorities for later architecture phases.

7.5.4 Applying Value Streams

Value streams provide valuable stakeholder context into why the organization needs business capabilities, while business capabilities provide what the organization needs for a particular value stage to be successful.

Start with the initial set of value stream models for the business documented in the Architecture Vision phase. Within the scope of the specific Enterprise Architecture project, if sufficiently larger in breadth, there may be a need for new value streams not already in the repository.

A new or existing value stream can be analyzed within the scope of the project through heat mapping (by value stream stage) or by developing use-cases around a complete definition of the value stream (see Baseline Example in the TOGAF® Series Guide: Value Streams). A project might focus on specific stakeholders, one element of business value, or stress some stages over others to develop better requirements for solutions in later phases.

The most substantive benefits come from mapping relationships between the stages in a value stream to business capabilities, then performing a gap analysis for capabilities (such as heat mapping) in the context of the business value achieved by the value stream for a specific stakeholder (see Mapping Value Streams to Business Capabilities in the TOGAF® Series Guide: Value Streams).

7.5.5 Applying the Organization Map

An organization map shows the key organizational units, partners, and stakeholder groups that make up the enterprise ecosystem. The map should also depict the working relationship between those entities, as distinct from an organizational chart that only shows hierarchical reporting relationships. The map is typically depicted as a network or web of relationships and interactions between the various business entities (see *Organigraphs: Drawing How Companies Really Work*, by Mintzberg and Van der Heyden, 1999).

The business unit is the main concept used to establish organization maps. In keeping with the relatively unconstrained view of what constitutes as enterprise, the enterprise may be one business unit for the project underway, may include all business units, or also include third parties or other stakeholder groups. The interpretation depends on the scope of the architecture effort.

This map is a key element of Business Architecture because it provides the organizational context for the whole Enterprise Architecture effort. While capability mapping exposes what a business does and value stream mapping exposes how it delivers value to specific stakeholders, the organization map identities the business units or third parties that possess or use those capabilities and which participate in the value streams.

Taken together with the methods in Section 7.5.3, Section 7.5.4, and the associated Guides, the organization map provides an understanding of which business units to involve in the architecture effort, who and when to talk about a given requirement, and how to measure the impact of various decisions.

7.5.6 Applying Modeling Techniques

The modeling and mapping techniques provided here are extensions that implement the business capabilities, value streams, and organization maps described above in Phase B into the practices of the business. They expand the operating model, which is a representation for how an organization operates across a range of domains in order to accomplish its function (see A Method for Identifying Process Re-Use Opportunities to Enhance the Operating Model, M. de Vries et al.).

In addition to the techniques described above (capability maps, value streams, and organization maps), a variety of other modeling techniques may be employed, if deemed appropriate. For example:

- **Activity Models** (also called **Business Process Models**) describe the functions associated with the enterprise's business activities, the data and/or information exchanged between activities (internal exchanges), and the data and/or information exchanged with other activities that are outside the scope of the model (external exchanges)

  Activity models are hierarchical in nature. They capture the activities performed in a business process, and the ICOMs (inputs, controls, outputs, and mechanisms/resources used) of those activities. Activity models can be annotated with explicit statements of business rules, which represent relationships among the ICOMs. For example, a business rule can specify who can do what under specified conditions, the combination of inputs and controls needed, and the resulting outputs. One technique for creating activity models is the IDEF (Integrated Computer Aided Manufacturing (ICAM) DEFinition) modeling technique.

  The Object Management Group (OMG) has developed the Business Process Modeling Notation™ (BPMN™), a standard for business process modeling that includes a language with which to specify business processes, their tasks/steps, and the documents produced.

- **Use-Case Models** can describe either business processes or systems functions, depending on the focus of the modeling effort

  A use-case model describes the business processes of an enterprise in terms of use-cases and actors corresponding to business processes and organizational participants (people, organizations, etc.). The use-case model is described in use-case diagrams and use-case specifications.

- **Class Models** are similar to logical data models

  A class model describes static information and relationships between information. A class model also describes informational behaviors. Like many of the other models, it can also be used to model various levels of granularity. Depending on the intent of the model, a class model can represent business domain entities or systems implementation classes. A business domain model represents key business information (domain classes), their characteristics (attributes), their behaviors (methods or operations), and relationships (often referred to as multiplicity, describing how many classes typically participate in the relationship), and cardinality (describes required or optional participation in the relationship). Specifications further elaborate and detail information that cannot be represented in the class diagram.

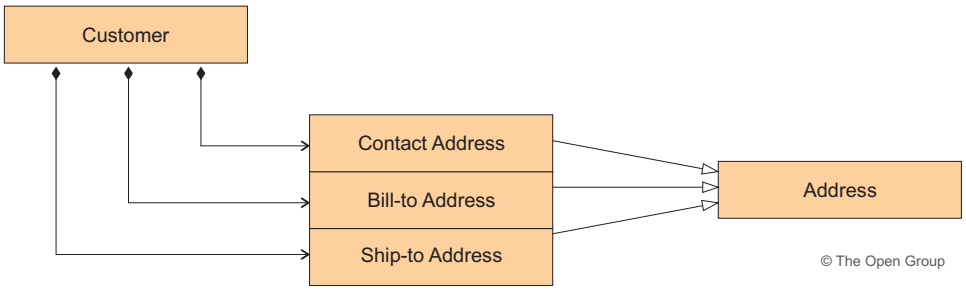

Figure 7-2 UML Business Class Diagram

All three types of model above can be represented in the Unified Modeling Language™ (UML®), and a variety of tools exist for generating such models.

Certain industry sectors have modeling techniques specific to the sector concerned. For example, the Defense sector uses the following models. These models have to be used carefully, especially if the location and conduct of business processes will be altered in the visionary Business Architecture.

- The **Node Connectivity Diagram** describes the business locations (nodes), the "needlines" between them, and the characteristics of the information exchanged

  Node connectivity can be described at three levels: conceptual, logical, and physical. Each needline indicates the need for some kind of information transfer between the two connected nodes. A node can represent a role (e.g., a CIO), an organizational unit, a business location or facility, and so on. An arrow indicating the direction of information flow is annotated to describe the characteristics of the data or information — for example, its content, media, security or classification level, timeliness, and requirements for information system interoperability.

- The **Information Exchange Matrix** documents the information exchange requirements for an Enterprise Architecture

  Information exchange requirements express the relationships across three basic entities (activities, business nodes and their elements, and information flow), and focus on characteristics of the information exchange, such as performance and security. They identify who exchanges what information with whom, why the information is necessary, and in what manner.

7.5.7 Architecture Repository

As part of Phase B, the architecture team will need to consider what relevant Business Architecture resources are available from the Architecture Repository (see Part V, Chapter 37), in particular:

- Industry reference models relevant to the organization's industry sector

  These are "Industry Architectures", in terms of the Enterprise Continuum. They are held in the Reference Library of the Architecture Repository (see Part V, Section 37.3). For example:

  - The Object Management Group (OMG) — www.omg.org — has a number of vertical Domain Task Forces developing industry reference models relevant to specific vertical domains such as Healthcare, Transportation, Finance, etc.
  - The TM Forum — www.tmforum.org — has developed detailed reference models relevant to the Telecommunications industry
  - Government departments and agencies in different countries have reference models and frameworks mandated for use, intended to promote cross-departmental integration and interoperability

    An example is the Federal Enterprise Architecture Business Reference Model, which is a function-driven framework for describing the business operations of the Federal Government independent of the agencies that perform them.
  - The IT4IT Reference Architecture provides a high-level IT Value Chain that can be used within the IT segment of your architecture

    The IT4IT Level 1 Reference Architecture can be used to guide the creation of a Business Capability Map for the IT segment.

- Enterprise-specific Business Architecture views (capability maps, value stream maps, organization maps, etc.)
- Enterprise-specific building blocks (process components, business rules, job descriptions, etc.)
- Applicable standards

Chapter 8

Phase C: Information Systems Architectures

This chapter describes the Information Systems Architectures for an architecture project, including the development of Data and Application Architectures.

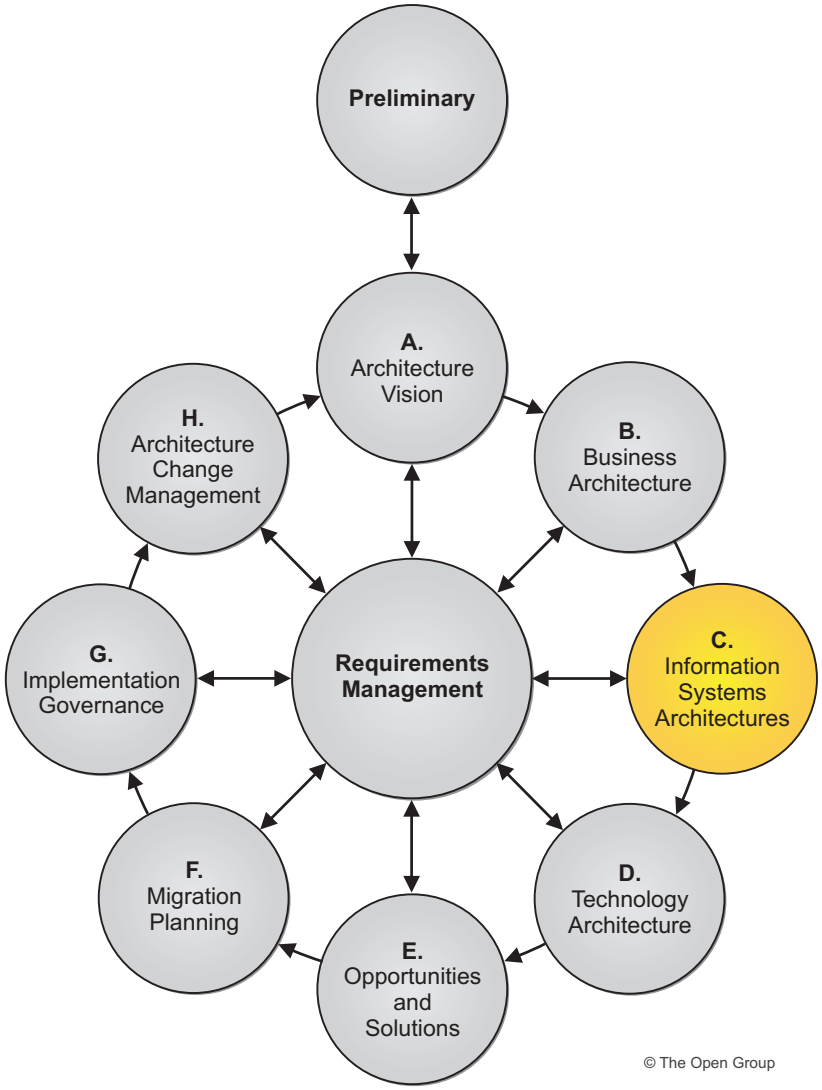

Figure 8-1 Phase C: Information Systems Architectures

8.1 Objectives

The objectives of Phase C are to:

- Develop the Target Information Systems Architectures, describing how the enterprise's Information Systems Architecture will enable the Business Architecture and the Architecture Vision, in a way that addresses the Statement of Architecture Work and stakeholder concerns
- Identify candidate Architecture Roadmap components based upon gaps between the Baseline and Target Information Systems (Data and Application) Architectures

8.2 Approach

Phase C involves some combination of Data and Application Architecture, in either order. Advocates exist for both sequences. For example, Steven Spewak's *Enterprise Architecture Planning (EAP)* recommends a data-driven approach.

On the other hand, major applications systems — such as those for Enterprise Resource Planning (ERP), Customer Relationship Management (CRM), etc. — often provide a combination of technology infrastructure and business application logic, and some organizations take an application-driven approach, whereby they recognize certain key applications as forming the core underpinning of the mission-critical business processes, and take the implementation and integration of those core applications as the primary focus of architecture effort (the integration issues often constituting a major challenge).

Detailed descriptions for Phase C are given separately for each architecture domain:

- Phase C: Information Systems Architectures — Data Architecture (see Chapter 9)
- Phase C: Information Systems Architectures — Application Architecture (see Chapter 10)

Chapter 9

Phase C: Information Systems Architectures — Data Architecture

This chapter describes the Data Architecture part of Phase C.

9.1 Objectives

The objectives of the Data Architecture part of Phase C are to:

- Develop the Target Data Architecture that enables the Business Architecture and the Architecture Vision, in a way that addresses the Statement of Architecture Work and stakeholder concerns
- Identify candidate Architecture Roadmap components based upon gaps between the Baseline and Target Data Architectures

9.2 Inputs

This section defines the inputs to Phase C (Data Architecture).

9.2.1 Reference Materials External to the Enterprise

- Architecture reference materials (see Part IV, Section 32.2.5)

9.2.2 Non-Architectural Inputs

- Request for Architecture Work (see Part IV, Section 32.2.17)
- Capability Assessment (see Part IV, Section 32.2.10)
- Communications Plan (see Part IV, Section 32.2.12)

9.2.3 Architectural Inputs

- Organizational Model for Enterprise Architecture (see Part IV, Section 32.2.16), including:
  - 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
- Tailored Architecture Framework (see Part IV, Section 32.2.21, on page 363), including:
  - Tailored architecture method
  - Tailored architecture content (deliverables and artifacts)
  - Configured and deployed tools
- Data principles (see Part III, Section 20.6.2), if existing
- Statement of Architecture Work (see Part IV, Section 32.2.20)
- Architecture Vision (see Part IV, Section 32.2.8)
- Architecture Repository (see Part IV, Section 32.2.5), including:
  - Re-usable building blocks (in particular, definitions of current data)
  - Publicly available reference models
  - Organization-specific reference models
  - Organization standards
- Draft Architecture Definition Document (see Part IV, Section 32.2.3), including:
  - Baseline Business Architecture, Version 1.0 (detailed), if appropriate
  - Target Business Architecture, Version 1.0 (detailed)
  - Baseline Data Architecture, Version 0.1, if available
  - Target Data Architecture, Version 0.1, if available
  - Baseline Application Architecture, Version 1.0 (detailed) or Version 0.1 (Vision)
  - Target Application Architecture, Version 1.0 (detailed) or Version 0.1 (Vision)
  - Baseline Technology Architecture, Version 0.1 (Vision)
  - Target Technology Architecture, Version 0.1 (Vision)
- Draft Architecture Requirements Specification (see Part IV, Section 32.2.6), including:
  - Gap analysis results (from Business Architecture)
  - Relevant technical requirements that will apply to this phase
- Business Architecture components of an Architecture Roadmap (see Part IV, Section 32.2.7)

9.3 Steps

The level of detail addressed in Phase C will depend on the scope and goals of the overall architecture effort.

New data building blocks being introduced as part of this effort will need to be defined in detail during Phase C. Existing data building blocks to be carried over and supported in the target environment may already have been adequately defined in previous architectural work; but, if not, they too will need to be defined in Phase C.

The order of the steps in this phase as well as the time at which they are formally started and completed should be adapted to the situation at hand in accordance with the established Architecture Governance. In particular, determine whether in this situation it is appropriate to conduct Baseline Description or Target Architecture development first, as described in Part III, Chapter 18.

All activities that have been initiated in these steps should be closed during the Finalize the Data Architecture step (see Section 9.3.8). The documentation generated from these steps must be formally published in the Create the Architecture Definition Document step (see Section 9.3.9).

The steps in Phase C (Data Architecture) are as follows:

- Select reference models, viewpoints, and tools (see Section 9.3.1)
- Develop Baseline Data Architecture Description (see Section 9.3.2)
- Develop Target Data Architecture Description (see Section 9.3.3)
- Perform gap analysis (see Section 9.3.4)
- Define candidate roadmap components (see Section 9.3.5)
- Resolve impacts across the Architecture Landscape (see Section 9.3.6)
- Conduct formal stakeholder review (see Section 9.3.7)
- Finalize the Data Architecture (see Section 9.3.8)
- Create the Architecture Definition Document (see Section 9.3.9)

9.3.1 Select Reference Models, Viewpoints, and Tools

Review and validate (or generate, if necessary) the set of data principles. These will normally form part of an overarching set of Architecture Principles. Guidelines for developing and applying principles, and a sample set of data principles, are given in Part III, Chapter 20.

Select relevant Data Architecture resources (reference models, patterns, etc.) on the basis of the business drivers, stakeholders, concerns, and Business Architecture.

Select relevant Data Architecture viewpoints (for example, stakeholders of the data — regulatory bodies, users, generators, subjects, auditors, etc.; various time dimensions — real-time, reporting period, event-driven, etc.; locations; business processes); i.e., those that will enable the architect to demonstrate how the stakeholder concerns are being addressed in the Data Architecture.

Identify appropriate tools and techniques (including forms) to be used for data capture, modeling, and analysis, in association with the selected viewpoints. Depending on the degree of sophistication warranted, these may comprise simple documents or spreadsheets, or more sophisticated modeling tools and techniques such as data management models, data models, etc.

Examples of data modeling techniques are:

- Entity-relationship diagram
- Class diagram

9.3.1.1 *Determine Overall Modeling Process*

For each viewpoint, select the models needed to support the specific view required, using the selected tool or method.

Ensure that all stakeholder concerns are covered. If they are not, create new models to address concerns not covered, or augment existing models (see above).

The recommended process for developing a Data Architecture is as follows:

- Collect data-related models from existing Business Architecture and Application Architecture materials
- Rationalize data requirements and align with any existing enterprise data catalogs and models; this allows the development of a data inventory and entity relationship
- Update and develop matrices across the architecture by relating data to business service, business function, access rights, and application
- Elaborate Data Architecture views by examining how data is created, distributed, migrated, secured, and archived

9.3.1.2 *Identify Required Catalogs of Data Building Blocks*

The organization's data inventory is captured as a catalog within the Architecture Repository. Catalogs are hierarchical in nature and capture a decomposition of a metamodel entity and also decompositions across related model entities (e.g., logical data component → physical data component → data entity).

Catalogs form the raw material for development of matrices and diagrams and also act as a key resource for managing the business and IT capability.

During the Business Architecture phase, a Business Service/Information diagram was created showing the key data entities required by the main business services. This is a prerequisite to successful Data Architecture activities.

Using the traceability from application to business function to data entity inherent in the content framework, it is possible to create an inventory of the data needed to be in place to support the Architecture Vision.

Once the data requirements are consolidated in a single location, it is possible to refine the data inventory to achieve semantic consistency and to remove gaps and overlaps.

The following catalogs should be considered for development within a Data Architecture:

- Data Entity/Data Component catalog

The structure of catalogs is based on the attributes of metamodel entities, as defined in Part IV, Chapter 30.

9.3.1.3 Identify Required Matrices

Matrices show the core relationships between related model entities.

Matrices form the raw material for development of diagrams and also act as a key resource for impact assessment.

At this stage, an entity to applications matrix could be produced to validate this mapping. How data is created, maintained, transformed, and passed to other applications, or used by other applications, will now start to be understood. Obvious gaps such as entities that never seem to be created by an application or data created but never used, need to be noted for later gap analysis.

The rationalized data inventory can be used to update and refine the architectural diagrams of how data relates to other aspects of the architecture.

Once these updates have been made, it may be appropriate to drop into a short iteration of the Application Architecture to resolve the changes identified.

The following matrices should be considered for development within a Data Architecture:

- Data Entity/Business Function (showing which data supports which functions and which business function owns which data)
- Business Service/Information (developed during the Business Architecture phase)
- Application/Data (developed across the Application Architecture and Data Architecture phases)

The structure of matrices is based on the attributes of metamodel entities, as defined in Part IV, Chapter 30.

9.3.1.4 Identify Required Diagrams

Diagrams present the Data Architecture information from a set of different perspectives (viewpoints) according to the requirements of the stakeholders.

Once the data entities have been refined, a diagram of the relationships between entities and their attributes can be produced.

It is important to note at this stage that information may be a mixture of enterprise-level data (from system service providers and package vendor information) and local-level data held in personal databases and spreadsheets.

The level of detail modeled needs to be carefully assessed. Some physical system data models will exist down to a very detailed level; others will only have core entities modeled. Not all data models will have been kept up-to-date as applications were modified and extended over time. It is important to achieve a balance in the level of detail provided (e.g., reproducing existing detailed system physical data schemas or presenting high-level process maps and data requirements, highlight the two extreme views).

The following diagrams should be considered for development within a Data Architecture:

- Conceptual Data diagram
- Logical Data diagram
- Data Dissemination diagram

- Data Lifecycle diagram
- Data Security diagram
- Data Migration diagram

9.3.1.5 *Identify Types of Requirement to be Collected*

Once the Data Architecture catalogs, matrices, and diagrams have been developed, architecture modeling is completed by formalizing the data-focused requirements for implementing the Target Architecture.

These requirements may:

- Relate to the data domain
- Provide requirements input into the Application and Technology Architectures
- Provide detailed guidance to be reflected during design and implementation to ensure that the solution addresses the original architecture requirements

Within this step, the architect should identify requirements that should be met by the architecture (see Section 16.5.2).

9.3.2 Develop Baseline Data Architecture Description

Develop a Baseline Description of the existing Data Architecture, to the extent necessary to support the Target Data Architecture. The scope and level of detail to be defined will depend on the extent to which existing data elements are likely to be carried over into the Target Data Architecture, and on whether architectural descriptions exist, as described in Section 9.5. To the extent possible, identify the relevant Data Architecture building blocks, drawing on the Architecture Repository (see Part V, Chapter 37).

Where new architecture models need to be developed to satisfy stakeholder concerns, use the models identified within Step 1 as a guideline for creating new architecture content to describe the Baseline Architecture.

9.3.3 Develop Target Data Architecture Description

Develop a Target Description for the Data Architecture, to the extent necessary to support the Architecture Vision and Target Business Architecture. The scope and level of detail to be defined will depend on the relevance of the data elements to attaining the Target Architecture, and on whether architectural descriptions exist. To the extent possible, identify the relevant Data Architecture building blocks, drawing on the Architecture Repository (see Part V, Chapter 37).

Where new architecture models need to be developed to satisfy stakeholder concerns, use the models identified within Step 1 as a guideline for creating new architecture content to describe the Target Architecture.

9.3.4 Perform Gap Analysis

Verify the architecture models for internal consistency and accuracy:

- Perform trade-off analysis to resolve conflicts (if any) among the different views
- Validate that the models support the principles, objectives, and constraints
- Note changes to the viewpoint represented in the selected models from the Architecture Repository, and document
- Test architecture models for completeness against requirements

Identify gaps between the Baseline and Target, using the gap analysis technique as described in Part III, Chapter 23.

9.3.5 Define Candidate Roadmap Components

Following the creation of a Baseline Architecture, Target Architecture, and gap analysis, a data roadmap is required to prioritize activities over the coming phases.

This initial Data Architecture roadmap will be used as raw material to support more detailed definition of a consolidated, cross-discipline roadmap within the Opportunities & Solutions phase.

9.3.6 Resolve Impacts Across the Architecture Landscape

Once the Data Architecture is finalized, it is necessary to understand any wider impacts or implications.

At this stage, other architecture artifacts in the Architecture Landscape should be examined to identify:

- Does this Data Architecture create an impact on any pre-existing architectures?
- Have recent changes been made that impact the Data Architecture?
- Are there any opportunities to leverage work from this Data Architecture in other areas of the organization?
- Does this Data Architecture impact other projects (including those planned as well as those currently in progress)?
- Will this Data Architecture be impacted by other projects (including those planned as well as those currently in progress)?

9.3.7 Conduct Formal Stakeholder Review

Check the original motivation for the architecture project and the Statement of Architecture Work against the proposed Data Architecture. Conduct an impact analysis to identify any areas where the Business and Application Architectures (e.g., business practices) may need to change to cater for changes in the Data Architecture (for example, changes to forms or procedures, applications, or database systems).

If the impact is significant, this may warrant the Business and Application Architectures being revisited.

Identify any areas where the Application Architecture (if generated at this point) may need to

change to cater for changes in the Data Architecture (or to identify constraints on the Application Architecture about to be designed).

If the impact is significant, it may be appropriate to drop into a short iteration of the Application Architecture at this point.

Identify any constraints on the Technology Architecture about to be designed, refining the proposed Data Architecture only if necessary.

9.3.8 Finalize the Data Architecture

- Select standards for each of the building blocks, re-using as much as possible from the reference models selected from the Architecture Repository
- Fully document each building block
- Conduct a final cross-check of overall architecture against business requirements; document the rationale for building block decisions in the architecture document
- Document the final requirements traceability report
- Document the final mapping of the architecture within the Architecture Repository; from the selected building blocks, identify those that might be re-used, and publish via the Architecture Repository
- Finalize all the work products, such as gap analysis

9.3.9 Create the Architecture Definition Document

Document the rationale for building block decisions in the Architecture Definition Document.

Prepare the Data Architecture sections of the Architecture Definition Document, comprising some or all of:

- Business data model
- Logical data model
- Data management process model
- Data Entity/Business Function matrix
- Data interoperability requirements (e.g., XML schema, security policies)
- If appropriate, use reports and/or graphics generated by modeling tools to demonstrate key views of the architecture; route the document for review by relevant stakeholders, and incorporate feedback

9.4 Outputs

The outputs of Phase C (Data Architecture) may include, but are not restricted to:

- Refined and updated versions of the Architecture Vision phase deliverables, where applicable:
  - Statement of Architecture Work (see Part IV, Section 32.2.20), updated if necessary

  - Validated data principles (see Part III, Section 20.6.2), or new data principles (if generated here)
- Draft Architecture Definition Document (see Part IV, Section 32.2.3), including:
  - Baseline Data Architecture, Version 1.0, if appropriate
  - Target Data Architecture, Version 1.0
    - Business data model
    - Logical data model
    - Data management process models
    - Data Entity/Business Function matrix
  - Views corresponding to the selected viewpoints addressing key stakeholder concerns
- Draft Architecture Requirements Specification (see Part IV, Section 32.2.6), including such Data Architecture requirements as:
  - Gap analysis results
  - Data interoperability requirements
  - Relevant technical requirements that will apply to this evolution of the architecture development cycle
  - Constraints on the Technology Architecture about to be designed
  - Updated business requirements, if appropriate
  - Updated application requirements, if appropriate
- Data Architecture components of an Architecture Roadmap (see Part IV, Section 32.2.7)

The outputs may include some or all of the following:

- Catalogs:
  - Data Entity/Data Component catalog
- Matrices:
  - Data Entity/Business Function matrix
  - Application/Data matrix
- Diagrams:
  - Conceptual Data diagram
  - Logical Data diagram
  - Data Dissemination diagram
  - Data Security diagram
  - Data Migration diagram
  - Data Lifecycle diagram

9.5 Approach

9.5.1 Key Considerations for Data Architecture

9.5.1.1 Data Management

When an enterprise has chosen to undertake largescale architectural transformation, it is important to understand and address data management issues. A structured and comprehensive approach to data management enables the effective use of data to capitalize on its competitive advantages.

Considerations include:

- A clear definition of which application components in the landscape will serve as the system of record or reference for enterprise master data
- Will there be an enterprise-wide standard that all application components, including software packages, need to adopt?

  (In the main, packages can be prescriptive about the data models and may not be flexible.)
- Clearly understand how data entities are utilized by business functions, processes, and services
- Clearly understand how and where enterprise data entities are created, stored, transported, and reported
- What is the level and complexity of data transformations required to support the information exchange needs between applications?
- What will be the requirement for software in supporting data integration with the enterprise's customers and suppliers (e.g., use of ETL tools during the data migration, data profiling tools to evaluate data quality, etc.)?

9.5.1.2 Data Migration

When an existing application is replaced, there will be a critical need to migrate data (master, transactional, and reference) to the new application. The Data Architecture should identify data migration requirements and also provide indicators as to the level of transformation, weeding, and cleansing that will be required to present data in a format that meets the requirements and constraints of the target application. The objective being that the target application has quality data when it is populated. Another key consideration is to ensure that an enterprise-wide common data definition is established to support the transformation.

9.5.1.3 Data Governance

Data governance considerations ensure that the enterprise has the necessary dimensions in place to enable the transformation, as follows:

- **Structure**: this dimension pertains to whether the enterprise has the necessary organizational structure and the standards bodies to manage data entity aspects of the transformation
- **Management System**: here enterprises should have the necessary management system and data-related programs to manage the governance aspects of data entities throughout its lifecycle

- **People**: this dimension addresses what data-related skills and roles the enterprise requires for the transformation

  If the enterprise lacks such resources and skills, the enterprise should consider either acquiring those critical skills or training existing internal resources to meet the requirements through a well-defined learning program.

9.5.2 Architecture Repository

As part of this phase, the architecture team will need to consider what relevant Data Architecture resources are available in the organization's Architecture Repository (see Part V, Chapter 37), in particular, generic data models relevant to the organization's industry "vertical" sector. For example:

- Energistics® — Data Exchange Standards for the Upstream Oil & Gas Industry
- National Information Exchange Model (US Government)
- The ARTS Operational Data Model and the ARTS Data Warehouse Model (Retail)

Chapter 10

Phase C: Information Systems Architectures — Application Architecture

This chapter describes the Application Architecture part of Phase C.

10.1 Objectives

The objectives of the Application Architecture part of Phase C are to:

- Develop the Target Application Architecture that enables the Business Architecture and the Architecture Vision, in a way that addresses the Statement of Architecture Work and stakeholder concerns
- Identify candidate Architecture Roadmap components based upon gaps between the Baseline and Target Application Architectures

10.2 Inputs

This section defines the inputs to Phase C (Application Architecture).

10.2.1 Reference Materials External to the Enterprise

- Architecture reference materials (see Part IV, Section 32.2.5)

10.2.2 Non-Architectural Inputs

- Request for Architecture Work (see Part IV, Section 32.2.17)
- Capability Assessment (see Part IV, Section 32.2.10)
- Communications Plan (see Part IV, Section 32.2.12)

10.2.3 Architectural Inputs

- Organizational Model for Enterprise Architecture (see Part IV, Section 32.2.16), including:
  - 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
- Tailored Architecture Framework (see Part IV, Section 32.2.21), including:
  - Tailored architecture method
  - Tailored architecture content (deliverables and artifacts)
  - Configured and deployed tools
- Application principles (see Part III, Section 20.6.3), if existing
- Statement of Architecture Work (see Part IV, Section 32.2.20)
- Architecture Vision (see Part IV, Section 32.2.8)
- Architecture Repository (see Part IV, Section 32.2.5), including:
  - Re-usable building blocks
  - Publicly available reference models
  - Organization-specific reference models
  - Organization standards
- Draft Architecture Definition Document (see Part IV, Section 32.2.3), including:
  - Baseline Business Architecture, Version 1.0 (detailed), if appropriate
  - Target Business Architecture, Version 1.0 (detailed)
  - Baseline Data Architecture, Version 1.0 (detailed), or Version 0.1 (Vision)
  - Target Data Architecture, Version 1.0 (detailed), or Version 0.1 (Vision)
  - Baseline Application Architecture, Version 0.1, if appropriate and if available
  - Target Application Architecture, Version 0.1, if available
  - Baseline Technology Architecture, Version 0.1 (Vision)
  - Target Technology Architecture, Version 0.1 (Vision)
- Draft Architecture Requirements Specification (see Part IV, Section 32.2.6), including:
  - Gap analysis results (from Business Architecture and Data Architecture, if available)
  - Relevant technical requirements that will apply to this phase
- Business and Data Architecture components of an Architecture Roadmap, if available (see Part IV, Section 32.2.7)

10.3 Steps

The level of detail addressed in Phase C will depend on the scope and goals of the overall architecture effort.

New application building blocks being introduced as part of this effort will need to be defined in detail during Phase C. Existing application building blocks to be carried over and supported in the target environment may already have been adequately defined in previous architectural work; but, if not, they too will need to be defined in Phase C.

The order of the steps in this phase as well as the time at which they are formally started and completed should be adapted to the situation at hand in accordance with the established Architecture Governance. In particular, determine whether in this situation it is appropriate to conduct Baseline Description or Target Architecture development first, as described in Part III, Chapter 18.

All activities that have been initiated in these steps should be closed during the Finalize the Application Architecture step (see Section 10.3.8). The documentation generated from these steps must be formally published in the Create the Architecture Definition Document step (see Section 10.3.9).

The steps in Phase C (Application Architecture) are as follows:

- Select reference models, viewpoints, and tools (see Section 10.3.1)
- Develop Baseline Application Architecture Description (see Section 10.3.2)
- Develop Target Application Architecture Description (see Section 10.3.3)
- Perform gap analysis (see Section 10.3.4)
- Define candidate roadmap components (see Section 10.3.5)
- Resolve impacts across the Architecture Landscape (see Section 10.3.6)
- Conduct formal stakeholder review (see Section 10.3.7)
- Finalize the Application Architecture (see Section 10.3.8)
- Create the Architecture Definition Document (see Section 10.3.9)

10.3.1 Select Reference Models, Viewpoints, and Tools

Review and validate (or generate, if necessary) the set of application principles. These will normally form part of an overarching set of Architecture Principles. Guidelines for developing and applying principles, and a sample set of application principles, are given in Part III, Chapter 20.

Select relevant Application Architecture resources (reference models, patterns, etc.) from the Architecture Repository, on the basis of the business drivers, the stakeholders, and their concerns.

Select relevant Application Architecture viewpoints (for example, stakeholders of the applications — viewpoints relevant to functional and individual users of applications, etc.); i.e., those that will enable the architect to demonstrate how the stakeholder concerns are being addressed in the Application Architecture.

Identify appropriate tools and techniques to be used for capture, modeling, and analysis, in association with the selected viewpoints. Depending on the degree of sophistication warranted, these may comprise simple documents or spreadsheets, or more sophisticated modeling tools and techniques.

Consider using platform-independent descriptions of business logic. For example, the OMG Model-Driven Architecture® (MDA®) offers an approach to modeling Application Architectures that preserves the business logic from changes to the underlying platform and implementation technology.

10.3.1.1 Determine Overall Modeling Process

For each viewpoint, select the models needed to support the specific view required, using the selected tool or method.

Ensure that all stakeholder concerns are covered. If they are not, create new models to address concerns not covered, or augment existing models (see above).

The recommended process for developing an Application Architecture is as follows:

- Understand the list of applications or application components that are required, based on the baseline Application Portfolio, what the requirements are, and the Business Architecture scope
- Simplify complicated applications by decomposing them into two or more applications
- Ensure that the set of application definitions is internally consistent, by removing duplicate functionality as far as possible, and combining similar applications into one
- Identify logical applications and the most appropriate physical applications
- Develop matrices across the architecture by relating applications to business service, business function, data, process, etc.
- Elaborate a set of Application Architecture views by examining how the application will function, capturing integration, migration, development, and operational concerns

The level and rigor of decomposition needed varies from enterprise to enterprise, as well as within an enterprise, and the architect should consider the enterprise's goals, objectives, scope, and purpose of the Enterprise Architecture effort to determine the level of decomposition.

The level of granularity should be sufficient to enable identification of gaps and the scope of candidate work packages.

10.3.1.2 Identify Required Catalogs of Application Building Blocks

The organization's Application Portfolio is captured as a catalog within the Architecture Repository. Catalogs are hierarchical in nature and capture a decomposition of a metamodel entity and also decompositions across related model entities (e.g., logical application component → physical application component → information system service).

Catalogs form the raw material for development of matrices and diagrams and also act as a key resource for managing the business and IT capability.

The structure of catalogs is based on the attributes of metamodel entities, as defined in Part IV, Chapter 30.

The following catalogs should be considered for development within an Application Architecture:

- Application Portfolio catalog
- Interface catalog

10.3.1.3 Identify Required Matrices

Matrices show the core relationships between related model entities.

Matrices form the raw material for development of diagrams and also act as a key resource for impact assessment.

Once the baseline Application Portfolio has been assembled, it is necessary to map the applications to their purpose in supporting the business. The initial mapping should focus on business services within the Business Architecture, as this is the level of granularity where architecturally significant decisions are most likely to be needed.

Once applications are mapped to business services, it will also be possible to make associations from applications to data, through the business-information diagrams developed during Business Architecture.

If readily available, baseline application data models may be used to validate the Business Architecture and also to identify which data is held locally and which is accessed remotely.

The Data Architecture phase will focus on these issues, so at this point it may be appropriate to drop into a short iteration of the Data Architecture if it is deemed to be valuable to the scope of the architecture engagement.

Using existing information in the baseline application catalog, the Application Architecture should identify user and organizational dependencies on applications. This activity will support future state planning by determining impacted user communities and also facilitating the grouping of applications by user type or user location.

A key user community to be specifically considered is the operational support organization. This activity should examine application dependencies on shared operations capabilities and produce a diagram on how each application is effectively operated and managed.

Specifically considering the needs of the operational community may identify requirements for new or extended governance capabilities and applications.

The following matrices should be considered for development within an Application Architecture:

- Application/Organization matrix
- Role/Application matrix
- Application Interaction matrix
- Application/Function matrix

The structure of matrices is based on the attributes of metamodel entities, as defined in Part IV, Chapter 30.

10.3.1.4 Identify Required Diagrams

Diagrams present the Application Architecture information from a set of different perspectives (viewpoints) according to the requirements of the stakeholders.

Once the desired functionality of an application is known, it is necessary to perform an internal assessment of how the application should be best structured to meet its requirements.

In the case of packaged applications, it is likely to be the case that the application supports a number of configuration options, add-on modules, or application services that may be applied to the solution. For custom developed applications, it is necessary to identify the high-level structure of the application in terms of modules or subsystems as a foundation to organize design activity.

The following diagrams should be considered for development within an Application Architecture:

- Application Communication diagram
- Application and User Location diagram
- Enterprise Manageability diagram
- Process/Application Realization diagram
- Application Migration diagram
- Software Distribution diagram
- Software Engineering diagram
- Application Use-Case diagram

The structure of diagrams is based on the attributes of metamodel entities, as defined in Part IV, Chapter 30.

10.3.1.5 Identify Types of Requirement to be Collected

Once the Application Architecture catalogs, matrices, and diagrams have been developed, architecture modeling is completed by formalizing the application-focused requirements for implementing the Target Architecture.

These requirements may:

- Relate to the application domain
- Provide requirements input into the Data and Technology Architectures
- Provide detailed guidance to be reflected during design and implementation to ensure that the solution addresses the original architecture requirements

Within this step, the architect should identify requirements that should be met by the architecture (see Section 16.5.2).

10.3.2 Develop Baseline Application Architecture Description

Develop a Baseline Description of the existing Application Architecture, to the extent necessary to support the Target Application Architecture. The scope and level of detail to be defined will depend on the extent to which existing applications are likely to be carried over into the Target Application Architecture, and on whether Architecture Descriptions exist, as described in Section 10.5. To the extent possible, identify the relevant Application Architecture building blocks, drawing on the Architecture Repository (see Part V, Chapter 37). If not already existing within the Architecture Repository, define each application in line with the Application Portfolio catalog (see Part IV, Chapter 30).

Where new architecture models need to be developed to satisfy stakeholder concerns, use the models identified within Step 1 as a guideline for creating new architecture content to describe the Baseline Architecture.

10.3.3 Develop Target Application Architecture Description

Develop a Target Description for the Application Architecture, to the extent necessary to support the Architecture Vision, Target Business Architecture, and Target Data Architecture. The scope and level of detail to be defined will depend on the relevance of the application elements to attaining the Target Architecture Vision, and on whether architectural descriptions exist. To the extent possible, identify the relevant Application Architecture building blocks, drawing on the Architecture Repository (see Part V, Chapter 37).

Where new architecture models need to be developed to satisfy stakeholder concerns, use the models identified within Step 1 as a guideline for creating new architecture content to describe the Target Architecture.

10.3.4 Perform Gap Analysis

Verify the architecture models for internal consistency and accuracy:

- Perform trade-off analysis to resolve conflicts (if any) among the different views
- Validate that the models support the principles, objectives, and constraints
- Note changes to the viewpoint represented in the selected models from the Architecture Repository, and document
- Test architecture models for completeness against requirements

Identify gaps between the baseline and target, using the gap analysis technique as described in Part III, Chapter 23.

10.3.5 Define Candidate Roadmap Components

Following the creation of a Baseline Architecture, Target Architecture, and gap analysis, an application roadmap is required to prioritize activities over the coming phases.

This initial Application Architecture roadmap will be used as raw material to support more detailed definition of a consolidated, cross-discipline roadmap within the Opportunities & Solutions phase.

10.3.6 Resolve Impacts Across the Architecture Landscape

Once the Application Architecture is finalized, it is necessary to understand any wider impacts or implications.

At this stage, other architecture artifacts in the Architecture Landscape should be examined to identify:

- Does this Application Architecture create an impact on any pre-existing architectures?
- Have recent changes been made that impact the Application Architecture?
- Are there any opportunities to leverage work from this Application Architecture in other areas of the organization?
- Does this Application Architecture impact other projects (including those planned as well as those currently in progress)?
- Will this Application Architecture be impacted by other projects (including those planned as well as those currently in progress)?

10.3.7 Conduct Formal Stakeholder Review

Check the original motivation for the architecture project and the Statement of Architecture Work against the proposed Application Architecture. Conduct an impact analysis, to identify any areas where the Business and Data Architectures (e.g., business practices) may need to change to cater for changes in the Application Architecture (for example, changes to forms or procedures, applications, or database systems). If the impact is significant, this may warrant the Business and Data Architectures being revisited.

Identify any constraints on the Technology Architecture (especially the infrastructure) about to be designed.

10.3.8 Finalize the Application Architecture

- Select standards for each of the building blocks, re-using as much as possible from the reference models selected from the Architecture Repository
- Fully document each building block
- Conduct a final cross-check of overall architecture against business requirements; document the rationale for building block decisions in the architecture document
- Document the final requirements traceability report
- Document the final mapping of the architecture within the Architecture Repository; from the selected building blocks, identify those that might be re-used, and publish via the Architecture Repository
- Finalize all the work products, such as gap analysis

10.3.9 Create the Architecture Definition Document

- Document the rationale for building block decisions in the Architecture Definition Document
- Prepare the Application Architecture sections of the Architecture Definition Document; if appropriate, use reports and/or graphics generated by modeling tools to demonstrate key views of the architecture; route the document for review by relevant stakeholders, and incorporate feedback

10.4 Outputs

The outputs of Phase C (Application Architecture) may include, but are not restricted to:

- Refined and updated versions of the Architecture Vision phase deliverables, where applicable:
  - Statement of Architecture Work (see Part IV, Section 32.2.20), updated if necessary
  - Validated application principles, or new application principles (if generated here)
- Draft Architecture Definition Document (see Part IV, Section 32.2.3), including:
  - Baseline Application Architecture, Version 1.0, if appropriate
  - Target Application Architecture, Version 1.0
  - Views corresponding to the selected viewpoints, addressing key stakeholder concerns
- Draft Architecture Requirements Specification (see Part IV, Section 32.2.6), including such Application Architecture requirements as:
  - Gap analysis results
  - Applications interoperability requirements
  - Relevant technical requirements that will apply to this evolution of the architecture development cycle
  - Constraints on the Technology Architecture about to be designed
  - Updated business requirements, if appropriate
  - Updated data requirements, if appropriate
- Application Architecture components of an Architecture Roadmap (see Part IV, Section 32.2.7)

The outputs may include some or all of the following:

- Catalogs:
  - Application Portfolio catalog
  - Interface catalog
- Matrices:
  - Application/Organization matrix
  - Role/Application matrix

  - — Application/Function matrix
  - — Application Interaction matrix
- Diagrams:
  - — Application Communication diagram
  - — Application and User Location diagram
  - — Application Use-Case diagram
  - — Enterprise Manageability diagram
  - — Process/Application Realization diagram
  - — Software Engineering diagram
  - — Application Migration diagram
  - — Software Distribution diagram

10.5 Approach

10.5.1 Architecture Repository

As part of this phase, the architecture team will need to consider what relevant Application Architecture resources are available in the Architecture Repository (see Part V, Chapter 37).

In particular:

- Generic business models relevant to the organization's industry "vertical" sector; for example:
  - — The Object Management Group (OMG) — www.omg.org — has a number of vertical Domain Task Forces developing software models relevant to specific vertical domains such as Healthcare, Transportation, Finance, etc.
  - — The Open Group has developed a detailed application architecture reference model for the IT segment of organizations (the IT4IT Reference Architecture)
  - — The TM Forum — www.tmforum.org — has developed detailed applications models relevant to the Telecommunications industry
- Application models relevant to common high-level business functions, such as electronic commerce, supply chain management, etc.

The Open Group has a Reference Model for Integrated Information Infrastructure (III-RM) — see the TOGAF® Series Guide: The TOGAF Integrated Information Infrastructure Reference Model (III-RM) — that focuses on the application-level components and services necessary to provide an integrated information infrastructure.

Chapter 11

Phase D: Technology Architecture

This chapter describes the development of a Technology Architecture for an architecture project.

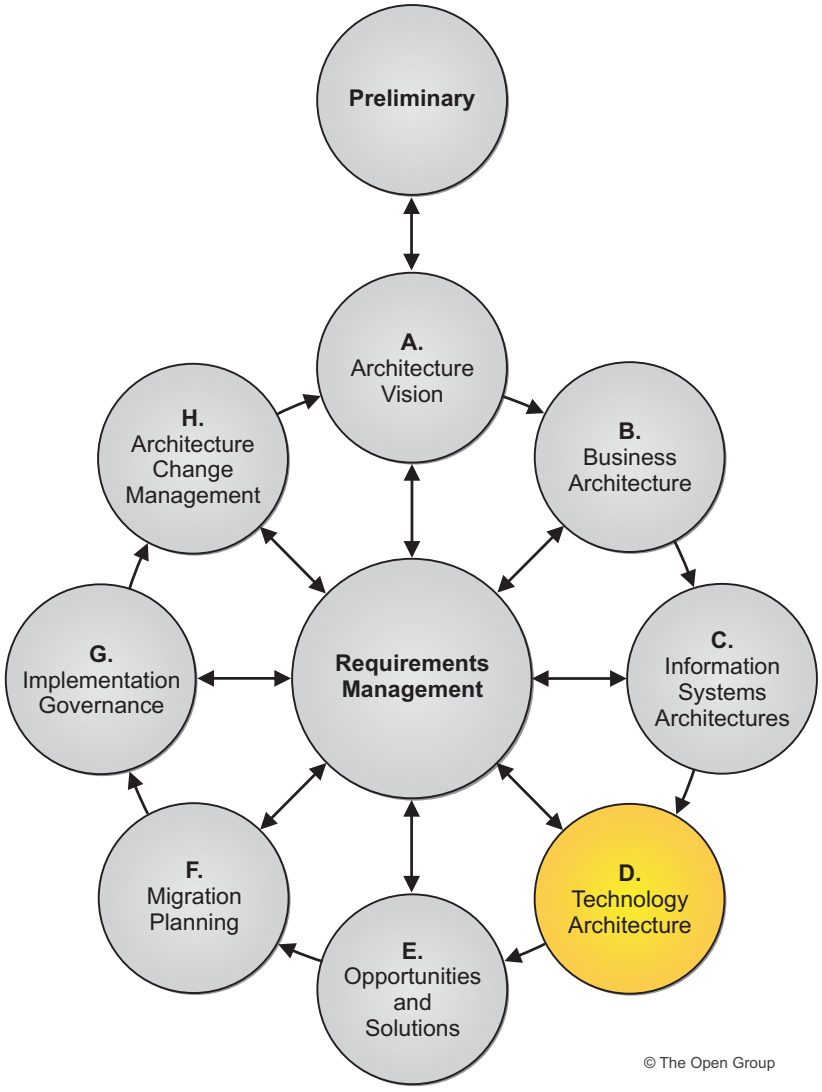

Figure 11-1 Phase D: Technology Architecture

11.1 Objectives

The objectives of Phase D are to:

- Develop the Target Technology Architecture that enables the Architecture Vision, target business, data, and application building blocks to be delivered through technology components and technology services, in a way that addresses the Statement of Architecture Work and stakeholder concerns
- Identify candidate Architecture Roadmap components based upon gaps between the Baseline and Target Technology Architectures

11.2 Inputs

This section defines the inputs to Phase D.

11.2.1 Reference Materials External to the Enterprise

- Architecture reference materials (see Part IV, Section 32.2.5)
- Product information on candidate products

11.2.2 Non-Architectural Inputs

- Request for Architecture Work (see Part IV, Section 32.2.17)
- Capability Assessment (see Part IV, Section 32.2.10)
- Communications Plan (see Part IV, Section 32.2.12)

11.2.3 Architectural Inputs

- Organizational Model for Enterprise Architecture (see Part IV, Section 32.2.16), including:
  - 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
- Tailored Architecture Framework (see Part IV, Section 32.2.21), including:
  - Tailored architecture method
  - Tailored architecture content (deliverables and artifacts)
  - Configured and deployed tools
- Technology principles (see Part III, Section 20.6.4), if existing

- Statement of Architecture Work (see Part IV, Section 32.2.20)
- Architecture Vision (see Part IV, Section 32.2.8)
- Architecture Repository (see Part IV, Section 32.2.5), including:
  - Re-usable building blocks
  - Publicly available reference models
  - Organization-specific reference models
  - Organization standards
- Draft Architecture Definition Document (see Part IV, Section 32.2.3), including:
  - Baseline Business Architecture, Version 1.0 (detailed)
  - Target Business Architecture Version 1.0 (detailed)
  - Baseline Data Architecture, Version 1.0 (detailed)
  - Target Data Architecture, Version 1.0 (detailed)
  - Baseline Application Architecture, Version 1.0 (detailed)
  - Target Application Architecture, Version 1.0 (detailed)
  - Baseline Technology Architecture, Version 0.1 (vision)
  - Target Technology Architecture, Version 0.1 (vision)
- Draft Architecture Requirements Specification (see Part IV, Section 32.2.6), including:
  - Gap analysis results (from Business, Data, and Application Architectures)
  - Relevant technical requirements from previous phases
- Business, Data, and Application Architecture components of an Architecture Roadmap (see Part IV, Section 32.2.7)

11.3 Steps

The level of detail addressed in Phase D will depend on the scope and goals of the overall architecture effort.

New technology building blocks being introduced as part of this effort will need to be defined in detail during Phase D. Existing technology building blocks to be supported in the target environment may need to be redefined in Phase D to ensure interoperability and fit-for-purpose within this specific Technology Architecture.

The order of the steps in Phase D as well as the time at which they are formally started and completed should be adapted to the situation at hand in accordance with the established Architecture Governance. In particular, determine whether in this situation it is appropriate to conduct Baseline Description or Target Architecture development first, as described in Part III, Chapter 18.

All activities that have been initiated in these steps should be closed during the Finalize the Technology Architecture step (see Section 11.3.8). The documentation generated from these steps must be formally published in the Create the Architecture Definition Document step (see Section 11.3.9).

The steps in Phase D are as follows:

- Select reference models, viewpoints, and tools (see Section 11.3.1)
- Develop Baseline Technology Architecture Description (see Section 11.3.2)
- Develop Target Technology Architecture Description (see Section 11.3.3)
- Perform gap analysis (see Section 11.3.4)
- Define candidate roadmap components (see Section 11.3.5)
- Resolve impacts across the Architecture Landscape (see Section 11.3.6)
- Conduct formal stakeholder review (see Section 11.3.7)
- Finalize the Technology Architecture (see Section 11.3.8)
- Create the Architecture Definition Document (see Section 11.3.9)

11.3.1 Select Reference Models, Viewpoints, and Tools

Review and validate the set of technology principles. These will normally form part of an overarching set of Architecture Principles. Guidelines for developing and applying principles, and a sample set of technology principles, are given in Part III, Chapter 20.

Select relevant Technology Architecture resources (reference models, patterns, etc.) from the Architecture Repository (see Part V, Chapter 37, on page 391), on the basis of the business drivers, stakeholders, and their concerns.

Select relevant Technology Architecture viewpoints that will enable the architect to demonstrate how the stakeholder concerns are being addressed in the Technology Architecture.

Identify appropriate tools and techniques to be used for capture, modeling, and analysis, in association with the selected viewpoints. Depending on the degree of sophistication required, these may comprise simple documents and spreadsheets, or more sophisticated modeling tools and techniques.

11.3.1.1 Determine Overall Modeling Process

For each viewpoint, select the models needed to support the specific view required, using the selected tool or method. Ensure that all stakeholder concerns are covered. If they are not, create new models to address them, or augment existing models (see above).

The process to develop a Technology Architecture incorporates the following steps:

- Define a taxonomy of technology services and logical technology components (including standards)
- Identify relevant locations where technology is deployed
- Carry out a physical inventory of deployed technology and abstract up to fit into the taxonomy
- Look at application and business requirements for technology
- Is the technology in place fit-for-purpose to meet new requirements (i.e., does it meet functional and non-functional requirements)?

  - Refine the taxonomy
  - Product selection (including dependent products)
- Determine configuration of the selected technology
- Determine impact:
  - Sizing and costing
  - Capacity planning
  - Installation/governance/migration impacts

In the earlier phases of the ADM, certain decisions made around service granularity and service boundaries will have implications on the technology component and the technology service. The areas where the Technology Architecture may be impacted will include the following:

- **Performance**: the granularity of the service will impact on technology service requirements

  Coarse-grained services contain several units of functionality with potentially varying non-functional requirements, so platform performance should be considered. In addition, coarse-grained services can sometimes contain more information than actually required by the requesting system.

- **Maintainability**: if service granularity is too coarse, then introducing changes to that service becomes difficult and impacts the maintenance of the service and the platform on which it is delivered

- **Location and Latency**: services might interact with each other over remote links and inter-service communication will have in-built latency

  Drawing service boundaries and setting the service granularity should consider platform/location impact of these inter-service communications.

- **Availability**: service invocation is subject to network and/or service failure

  So high communication availability is an important consideration during service decomposition and defining service granularity

Product selection processes may occur within the Technology Architecture phase where existing products are re-used, incremental capacity is being added, or product selection decisions are a constraint during project initiation.

Where product selection deviates from existing standards, involves significant effort, or has wide-ranging impact, this activity should be flagged as an opportunity and addressed through the Opportunities & Solutions phase.

11.3.1.2 Identify Required Catalogs of Technology Building Blocks

Catalogs are inventories of the core assets of the business. Catalogs are hierarchical in nature and capture a decomposition of a metamodel entity and also decompositions across related model entities (e.g., technology service → logical technology component → physical technology component).

Catalogs form the raw material for development of matrices and diagrams and also act as a key resource for managing the business and IT capability.

The Technology Architecture should create technology catalogs as follows:

- Based on existing technology catalogs and analysis of applications carried out in the Application Architecture phase, collect a list of products in use
- If the requirements identified in the Application Architecture are not met by existing products, extend the product list by examining products available on the market that provide the functionality and meet the required standards
- Classify products against the selected taxonomy if appropriate, extending the model as necessary to fit the classification of technology products in use
- If technology standards are currently in place, apply these to the technology component catalog to gain a baseline view of compliance with technology standards

The following catalogs should be considered for development within a Technology Architecture:

- Technology standards
- Technology portfolio

The structure of catalogs is based on the attributes of metamodel entities, as defined in Part IV, Chapter 30.

11.3.1.3 Identify Required Matrices

Matrices show the core relationships between related model entities.

Matrices form the raw material for development of diagrams and also act as a key resource for impact assessment.

The following matrix should be considered for development within a Technology Architecture:

- Application/Technology matrix

11.3.1.4 Identify Required Diagrams

Diagrams present the Technology Architecture information from a set of different perspectives (viewpoints) according to the requirements of the stakeholders.

This activity provides a link between platform requirements and hosting requirements, as a single application may need to be physically located in several environments to support local access, development lifecycles, and hosting requirements.

For major baseline applications or application platforms (where multiple applications are hosted on the same infrastructure stack), produce a stack diagram showing how hardware, operating system, software infrastructure, and packaged applications combine.

If appropriate, extend the Application Architecture diagrams of software distribution to show how applications map onto the technology platform.

For each environment, produce a logical diagram of hardware and software infrastructure showing the contents of the environment and logical communications between components. Where available, collect capacity information on the deployed infrastructure.

For each environment, produce a physical diagram of communications infrastructure, such as routers, switches, firewalls, and network links. Where available, collect capacity information on the communications infrastructure.

The following diagrams should be considered for development within a Technology Architecture:

- Environments and Locations diagram
- Platform Decomposition diagram
- Processing diagram
- Networked Computing/Hardware diagram
- Network and Communications diagram

The structure of diagrams is based on the attributes of metamodel entities, as defined in Part IV, Chapter 30.

11.3.1.5 Identify Types of Requirement to be Collected

Once the Technology Architecture catalogs, matrices, and diagrams have been developed, architecture modeling is completed by formalizing the technology-focused requirements for implementing the Target Architecture.

These requirements may:

- Relate to the technology domain
- Provide detailed guidance to be reflected during design and implementation to ensure that the solution addresses the original architecture requirements

Within this step, the architect should identify requirements that should be met by the architecture (see Section 16.5.2).

11.3.1.6 Select Services

The services portfolios are combinations of basic services from the service categories in the defined taxonomy that do not conflict. The combination of services are again tested to ensure support for the applications. This is a prerequisite to the later step of defining the architecture fully.

The previously identified requirements can provide more detailed information about:

- Requirements for organization-specific elements or pre-existing decisions (as applicable)
- Pre-existing and unchanging organizational elements (as applicable)
- Inherited external environment constraints

Where requirements demand definition of specialized services that are not identified in the TOGAF standard, consideration should be given to how these might be replaced if standardized services become available in the future.

For each building block, build up a service description portfolio as a set of non-conflicting services. The set of services must be tested to ensure that the functionality provided meets application requirements.

11.3.2 Develop Baseline Technology Architecture Description

Develop a Baseline Description of the existing Technology Architecture, to support the Target Technology Architecture. The scope and level of detail to be defined will depend on the extent to which existing technology components are likely to be carried over into the Target Technology Architecture, and on whether architectural descriptions exist, as described in Section 11.5.

Identify the relevant Technology Architecture building blocks, drawing on any artifacts held in the Architecture Repository. If nothing exists within the Architecture Repository, define each application in line with the Technology Portfolio catalog (see Part IV, Chapter 30).

Begin by converting the description of the existing environment into the terms of the organization's taxonomy of technology services and technology components (e.g., the TOGAF TRM). This will allow the team developing the architecture to gain experience and understanding of the taxonomy. The team may be able to take advantage of a previous architectural definition, but it is assumed that some adaptation may be required to match the architectural definition techniques described as part of this process. Another important task is to set down a list of key questions which can be used later in the development process to measure the effectiveness of the new architecture.

Where new architecture models need to be developed to satisfy stakeholder concerns, use the models identified within Step 1 as a guideline for creating new architecture content to describe the Baseline Architecture.

11.3.3 Develop Target Technology Architecture Description

Develop a Target Description for the Technology Architecture, to the extent necessary to support the Architecture Vision, Target Business Architecture, and Target Information Systems Architecture. The scope and level of detail to be defined will depend on the relevance of the technology elements to attaining the Target Architecture, and on whether architectural descriptions exist. To the extent possible, identify the relevant Technology Architecture building blocks, drawing on the Architecture Repository (see Part V, Chapter 37).

A key process in the creation of a broad architectural model of the target system is the conceptualization of building blocks. Architecture Building Blocks (ABBs) describe the functionality and how they may be implemented without the detail introduced by configuration or detailed design. The method of defining building blocks, along with some general guidelines for their use in creating an architectural model, is described in Part IV, Section 33.3.

Where new architecture models need to be developed to satisfy stakeholder concerns, use the models identified within Step 1 as a guideline for creating new architecture content to describe the Target Architecture.

11.3.4 Perform Gap Analysis

Verify the architecture models for internal consistency and accuracy:

- Perform trade-off analysis to resolve conflicts (if any) among the different views
- Validate that the models support the principles, objectives, and constraints
- Note changes to the viewpoint represented in the selected models from the Architecture Repository, and document

- Test architecture models for completeness against requirements

Identify gaps between the baseline and target, using the gap analysis technique as described in Part III, Chapter 23.

11.3.5 Define Candidate Roadmap Components

Following the creation of a Baseline Architecture, Target Architecture, and gap analysis, a Technology Roadmap is required to prioritize activities over the coming phases.

This initial Technology Architecture roadmap will be used as raw material to support more detailed definition of a consolidated, cross-discipline roadmap within the Opportunities & Solutions phase.

11.3.6 Resolve Impacts Across the Architecture Landscape

Once the Technology Architecture is finalized, it is necessary to understand any wider impacts or implications.

At this stage, other architecture artifacts in the Architecture Landscape should be examined to identify:

- Does this Technology Architecture create an impact on any pre-existing architectures?
- Have recent changes been made that impact the Technology Architecture?
- Are there any opportunities to leverage work from this Technology Architecture in other areas of the organization?
- Does this Technology Architecture impact other projects (including those planned as well as those currently in progress)?
- Will this Technology Architecture be impacted by other projects (including those planned as well as those currently in progress)?

11.3.7 Conduct Formal Stakeholder Review

Check the original motivation for the architecture project and the Statement of Architecture Work against the proposed Technology Architecture, asking if it is fit for the purpose of supporting subsequent work in the other architecture domains. Refine the proposed Technology Architecture only if necessary.

11.3.8 Finalize the Technology Architecture

- Select standards for each of the building blocks, re-using as much as possible from the reference models selected from the Architecture Repository
- Fully document each building block
- Conduct a final cross-check of overall architecture against business goals; document the rationale for building block decisions in the architecture document
- Document the final requirements traceability report

- Document the final mapping of the architecture within the Architecture Repository; from the selected building blocks, identify those that might be re-used (working practices, roles, business relationships, job descriptions, etc.), and publish via the Architecture Repository
- Finalize all the work products, such as gap analysis

11.3.9 Create the Architecture Definition Document

Document the rationale for building block decisions in the Architecture Definition Document.

Prepare the technology sections of the Architecture Definition Document, comprising some or all of:

- Fundamental functionality and attributes — semantic, unambiguous including security capability and manageability
- Dependent building blocks with required functionality and named interfaces
- Interfaces — chosen set, supplied (APIs, data formats, protocols, hardware interfaces, standards)
- Map to business/organizational entities and policies

If appropriate, use reports and/or graphics generated by modeling tools to demonstrate key views of the architecture. Route the document for review by relevant stakeholders, and incorporate feedback.

11.4 Outputs

The outputs of Phase D may include, but are not restricted to:

- Refined and updated versions of the Architecture Vision phase deliverables, where applicable:
  - Statement of Architecture Work (see Part IV, Section 32.2.20), updated if necessary
  - Validated technology principles, or new technology principles (if generated here)
- Draft Architecture Definition Document (see Part IV, Section 32.2.3), including:
  - Target Technology Architecture, Version 1.0 (detailed), including:
    - Technology Components and their relationships to information systems
    - Technology platforms and their decomposition, showing the combinations of technology required to realize a particular technology "stack"
    - Environments and locations — a grouping of the required technology into computing environments (e.g., development, production)
    - Expected processing load and distribution of load across technology components
    - Physical (network) communications
    - Hardware and network specifications

  - Baseline Technology Architecture, Version 1.0 (detailed), if appropriate
  - Views corresponding to the selected viewpoints addressing key stakeholder concerns
- Draft Architecture Requirements Specification (see Part IV, Section 32.2.6), including such Technology Architecture requirements as:
  - Gap analysis results
  - Requirements output from Phases B and C
  - Updated technology requirements
- Technology Architecture components of an Architecture Roadmap (see Part IV, Section 32.2.7)

The outputs may include some or all of the following:

- Catalogs:
  - Technology Standards catalog
  - Technology Portfolio catalog
- Matrices:
  - Application/Technology matrix
- Diagrams:
  - Environments and Locations diagram
  - Platform Decomposition diagram
  - Processing diagram
  - Networked Computing/Hardware diagram
  - Network and Communications diagram

11.5 Approach

11.5.1 Emerging Technologies

The evolution of new technologies is a major driver for change in enterprises looking for new innovative ways of operating and improving their business. The Technology Architecture needs to capture the transformation opportunities available to the enterprise through the adoption of new technology.

While the Enterprise Architecture is led by the business concerns, drivers for change are often found within evolving technology capabilities. As more digital innovations reach the market, stakeholders need to both anticipate and be open to technology-driven change. Part of Digital Transformation has arisen due to the convergence of telecommunications and computer capabilities which have opened up new ways of implementing infrastructures.

Solution development methods are also evolving to challenge traditional development methods and putting pressure on the shared services and common use benefits of the traditional Enterprise Architecture approach. Yet without a strong Enterprise Architecture approach, the rapid adoption of changing technologies will cause discontinuities across the enterprise.

The flexibility of the TOGAF ADM enables technology change to become a driver and strategic resource rather than a recipient of Change Requests. As a result, the Technology Architecture may both drive business capabilities and respond to information system requirements at the same time.

11.5.2 Architecture Repository

As part of Phase D, the architecture team will need to consider what relevant Technology Architecture resources are available in the Architecture Repository (see Part V, Chapter 37).

In particular:

- Existing IT services as documented in the IT repository or IT service catalog
- The adopted technical reference model, if applicable
- Generic technology models relevant to the organization's industry "vertical" sector; for example:
  - The TM Forum — www.tmforum.org — has developed detailed technology models relevant to the Telecommunications industry
- Technology models relevant to Common Systems Architectures
  - The Open Group has a Reference Model for Integrated Information Infrastructure (III-RM) — see the TOGAF® Series Guide: The TOGAF Integrated Information Infrastructure Reference Model (III-RM) — that focuses on the application-level components and underlying services necessary to provide an integrated information infrastructure

Chapter 12

Phase E: Opportunities & Solutions

This chapter describes the process of identifying delivery vehicles (projects, programs, or portfolios) that effectively deliver the Target Architecture identified in previous phases.

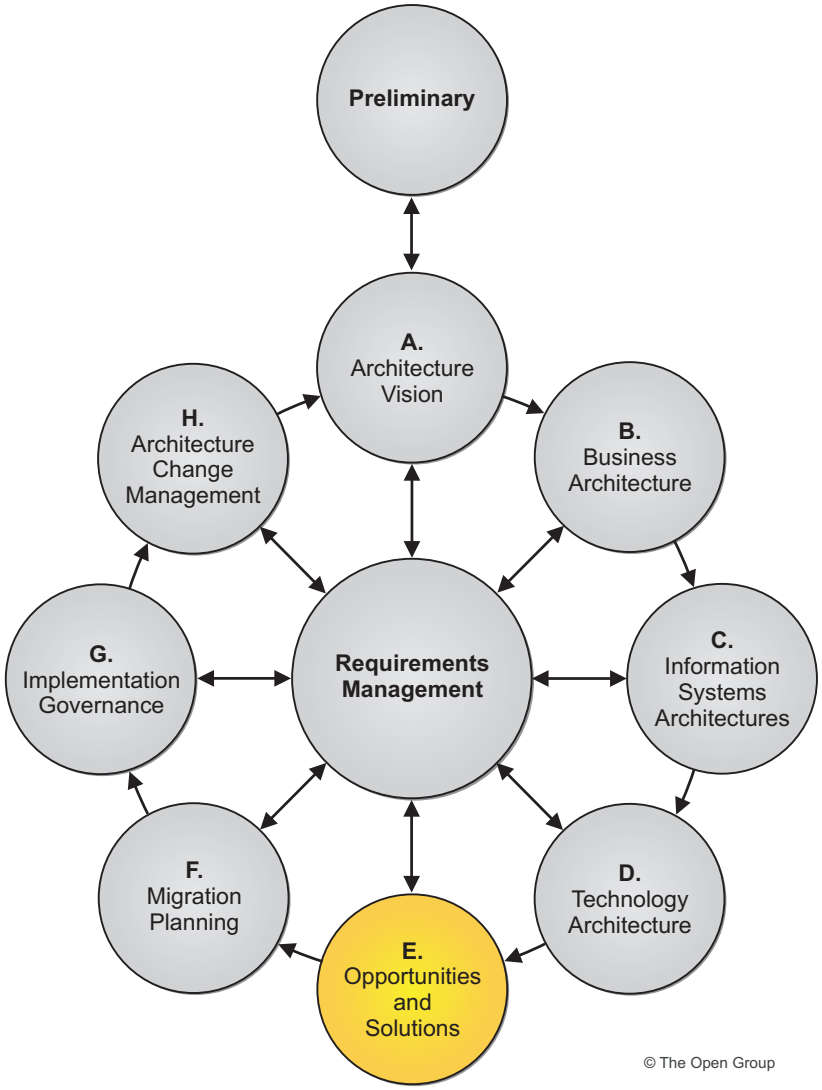

Figure 12-1 Phase E: Opportunities & Solutions

12.1 Objectives

The objectives of Phase E are to:

- Generate the initial complete version of the Architecture Roadmap, based upon the gap analysis and candidate Architecture Roadmap components from Phases B, C, and D
- Determine whether an incremental approach is required, and if so identify Transition Architectures that will deliver continuous business value
- Define the overall solution building blocks to finalize the Target Architecture based on the Architecture Building Blocks (ABBs)

12.2 Inputs

This section defines the inputs to Phase E.

12.2.1 Reference Materials External to the Enterprise

- Architecture reference materials (see Part IV, Section 32.2.5)
- Product information

12.2.2 Non-Architectural Inputs

- Request for Architecture Work (see Part IV, Section 32.2.17)
- Capability Assessment (see Part IV, Section 32.2.10)
- Communications Plan (see Part IV, Section 32.2.12)
- Planning methodologies

12.2.3 Architectural Inputs

- Organizational Model for Enterprise Architecture (see Part IV, Section 32.2.16), including:
  - 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
- Governance models and frameworks for:
  - Corporate Business Planning
  - Enterprise Architecture
  - Portfolio, Program, Project Management

  - System Development/Engineering
  - Operations (Service)
- Tailored Architecture Framework (see Part IV, Section 32.2.21), including:
  - Tailored architecture method
  - Tailored architecture content (deliverables and artifacts)
  - Configured and deployed tools
- Statement of Architecture Work (see Part IV, Section 32.2.20)
- Architecture Vision (see Part IV, Section 32.2.8)
- Architecture Repository (see Part IV, Section 32.2.5), including:
  - Re-usable building blocks
  - Publicly available reference models
  - Organization-specific reference models
  - Organization standards
- Draft Architecture Definition Document (see Part IV, Section 32.2.3), including:
  - Baseline Business Architecture, Version 1.0 (detailed)
  - Target Business Architecture, Version 1.0 (detailed)
  - Baseline Data Architecture, Version 1.0 (detailed)
  - Target Data Architecture, Version 1.0 (detailed)
  - Baseline Application Architecture, Version 1.0 (detailed)
  - Target Application Architecture, Version 1.0 (detailed)
  - Baseline Technology Architecture, Version 1.0 (detailed)
  - Target Technology Architecture, Version 1.0 (detailed)
- Draft Architecture Requirements Specification (see Part IV, Section 32.2.6), including:
  - Architectural requirements
  - Gap analysis results (from Business, Data, Application, and Technology Architecture)
  - IT Service Management requirements
- Change Requests for existing business programs and projects (see Part IV, Section 32.2.11)
- Candidate Architecture Roadmap components from Phases B, C, and D

12.3 Steps

The level of detail addressed in Phase E will depend on the scope and goals of the overall architecture effort.

The order of the steps in Phase E as well as the time at which they are formally started and completed should be adapted to the situation at hand in accordance with the established Architecture Governance.

All activities that have been initiated in these steps must be closed during the Create the Architecture Roadmap & Implementation and Migration Plan step (see Section 12.3.11).

The steps in Phase E are as follows:

- Determine/confirm key corporate change attributes (see Section 12.3.1)
- Determine business constraints for implementation (see Section 12.3.2)
- Review and consolidate gap analysis results from Phases B to D (see Section 12.3.3)
- Review consolidated requirements across related business functions (see Section 12.3.4)
- Consolidate and reconcile interoperability requirements (see Section 12.3.5)
- Refine and validate dependencies (see Section 12.3.6)
- Confirm readiness and risk for business transformation (see Section 12.3.7)
- Formulate Implementation and Migration Strategy (see Section 12.3.8)
- Identify and group major work packages (see Section 12.3.9)
- Identify Transition Architectures (see Section 12.3.10)
- Create the Architecture Roadmap & Implementation and Migration Plan (see Section 12.3.11)

12.3.1 Determine/Confirm Key Corporate Change Attributes

This step determines how the Enterprise Architecture can be best implemented to take advantage of the organization's business culture. This should include the creation of an Implementation Factor Assessment and Deduction matrix (see Part III, Section 24.1) to serve as a repository for architecture implementation and migration decisions. The step also includes assessments of the transition capabilities of the organization units involved (including culture and abilities), and assessments of the enterprise (including culture and skill sets).

The resulting factors from the assessments should be documented in the Implementation Factor Assessment and Deduction matrix. For organizations where Enterprise Architecture is well established, this step can be simple, but the matrix has to be established so that it can be used as an archive and record of decisions taken.

12.3.2 Determine Business Constraints for Implementation

Identify any business drivers that would constrain the sequence of implementation. This should include a review of the business and strategic plans, at both a corporate and line-of-business level, and a review of the Enterprise Architecture Maturity Assessment.

12.3.3 Review and Consolidate Gap Analysis Results from Phases B to D

Consolidate and integrate the gap analysis results from the Business, Information Systems, and Technology Architectures (created in Phases B to D) and assess their implications with respect to potential solutions and inter-dependencies. This should be done by creating a Consolidated Gaps, Solutions, and Dependencies matrix, as shown in Part III, Section 24.2, which will enable the identification of Solution Building Blocks (SBBs) that could potentially address one or more gaps and their associated ABBs.

Review the Phase B, C, and D gap analysis results and consolidate them in a single list. The gaps should be consolidated along with potential solutions to the gaps and dependencies. A recommended technique for determining the dependencies is to use sets of views such as the

Business Interaction matrix, the Data Entity/Business Function matrix, and the Application/Function matrix to completely relate elements from different architectural domains.

Rationalize the Consolidated Gaps, Solutions, and Dependencies matrix. Once all of the gaps have been documented, re-organize the gap list and place similar items together. When grouping the gaps, refer to the Implementation Factor Assessment and Deduction matrix and review the implementation factors. Any additional factors should be added to the Implementation Factor Assessment and Deduction matrix.

12.3.4 Review Consolidated Requirements Across Related Business Functions

Assess the requirements, gaps, solutions, and factors to identify a minimal set of requirements whose integration into work packages would lead to a more efficient and effective implementation of the Target Architecture across the business functions that are participating in the architecture. This functional perspective leads to the satisfaction of multiple requirements through the provision of shared solutions and services. The implications of this consolidation of requirements with respect to architectural components can be significant with respect to the provision of resources. For example, several requirements raised by several lines of business can be resolved through the provision of a shared set of Business Services and Information System Services within a work package or project.

12.3.5 Consolidate and Reconcile Interoperability Requirements

Consolidate the interoperability requirements identified in previous phases. The Architecture Vision and Target Architectures, as well as the Implementation Factor Assessment and Deduction matrix and Consolidated Gaps, Solutions, and Dependencies matrix, should be consolidated and reviewed to identify any constraints on interoperability required by the potential set of solutions.

A key outcome is to minimize interoperability conflicts, or to ensure such conflicts are addressed in the architecture. Re-used SBBs, Commercial Off-The-Shelf (COTS) products, and third-party service providers typically impose interoperability requirements that conflict. Any such conflicts must be addressed in the architecture, and conflicts must be considered across all architecture domains (Business, Applications, Data, and Technology).

There are two basic approaches to interoperability conflicts; either create a building block that transforms or translates between conflicting building blocks, or make a change to the specification of the conflicting building blocks.

12.3.6 Refine and Validate Dependencies

Refine the initial dependencies, ensuring that any constraints on the Implementation and Migration Plans are identified. There are several key dependencies that should be taken into account, such as dependencies on existing implementations of Business Services and Information System Services or changes to them. Dependencies should be used for determining the sequence of implementation and identifying the co-ordination required. A study of the dependencies should group activities together, creating a basis for projects to be established. Examine the relevant projects and see whether logical increments of deliverables can be identified. The dependencies will also help to identify when the identified increments can be delivered. Once finished, an assessment of these dependencies should be documented as part of the Architecture Roadmap and any necessary Transition Architectures.

Addressing dependencies serves as the basis for most migration planning.

12.3.7 Confirm Readiness and Risk for Business Transformation

Review the findings of the Business Transformation Readiness Assessment previously conducted in Phase A and determine their impact on the Architecture Roadmap and the Implementation and Migration Strategy. It is important to identify, classify, and mitigate risks associated with the transformation effort. Risks should be documented in the Consolidated Gaps, Solutions, and Dependencies matrix.

12.3.8 Formulate Implementation and Migration Strategy

Create an overall Implementation and Migration Strategy that will guide the implementation of the Target Architecture, and structure any Transition Architectures. The first activity is to determine an overall strategic approach to implementing the solutions and/or exploiting opportunities. There are three basic approaches as follows:

- Greenfield: a completely new implementation
- Revolutionary: a radical change (i.e., switch on, switch off)
- Evolutionary: a strategy of convergence, such as parallel running or a phased approach to introduce new capabilities

Next, determine an approach for the overall strategic direction that will address and mitigate the risks identified in the Consolidated Gaps, Solutions, and Dependencies matrix. The most common implementation methodologies are:

- Quick win (snapshots)
- Achievable targets
- Value chain method

These approaches and the identified dependencies should become the basis for the creation of the work packages. This activity terminates with agreement on the Implementation and Migration Strategy for the enterprise.

12.3.9 Identify and Group Major Work Packages

Key stakeholders, planners, and the Enterprise Architects should assess the missing business capabilities identified in the Architecture Vision and Target Architecture.

Using the Consolidated Gaps, Solutions, and Dependencies matrix together with the Implementation Factor Assessment and Deduction matrix, logically group the various activities into work packages.

Fill in the "Solution" column in the Consolidated Gaps, Solutions, and Dependencies matrix to recommend the proposed solution mechanisms. Indicate for every gap/activity whether the solution should be oriented towards a new development, or be based on an existing product, and/or use a solution that can be purchased. An existing system may resolve the requirement with minor enhancements. For new development this is a good time to determine whether the work should be conducted in-house or through a contract.

Classify every current system that is under consideration as:

- Mainstream: part of the future information system

- Contain: expected to be replaced or modified in the planning horizon (next three years)
- Replace: to be replaced in the planning horizon

Supporting top-level work packages should then in turn be decomposed into increments to deliver the capability increments. Analyze and refine these work packages or increments with respect to their business transformation issues and the strategic implementation approach. Finally, group the work packages into portfolios and projects within a portfolio, taking into consideration the dependencies and the strategic implementation approach.

12.3.10 Identify Transition Architectures

Where the scope of change to implement the Target Architecture requires an incremental approach, then one or more Transition Architectures may be necessary. These provide an ability to identify clear targets along the roadmap to realizing the Target Architecture. The Transition Architectures should provide measurable business value. The time-span between successive Transition Architectures does not have to be of uniform duration.

Development of Transition Architectures must be based upon the preferred implementation approach, the Consolidated Gaps, Solutions, and Dependencies matrix, the listing of projects and portfolios, as well as the enterprise's capacity for creating and absorbing change.

Determine where the difficult activities are, and unless there are compelling reasons, implement them after other activities that most easily deliver missing capability.

12.3.11 Create the Architecture Roadmap & Implementation and Migration Plan

Consolidate the work packages and Transition Architectures into the Architecture Roadmap, Version 0.1, which describes a timeline of the progression from the Baseline Architecture to the Target Architecture. The timeline informs the Implementation and Migration Plan. The Architecture Roadmap frames the migration planning in Phase F. Identified Transition Architectures and work packages should have a clear set of outcomes. The Architecture Roadmap must demonstrate how the selection and timeline of Transition Architectures and work packages realizes the Target Architecture.

The detail of the Architecture Roadmap, Version 0.1 should be expressed at a similar level of detail to the Architecture Definition Document developed in Phases B, C, and D. Where significant additional detail is required before implementation the architecture is likely transitioning to a different level. See Part III, Chapter 18 and Chapter 19 for techniques to manage iteration and different levels of detail.

The Implementation and Migration Plan must demonstrate the activity necessary to realize the Architecture Roadmap. The Implementation and Migration Plan forms the basis of the migration planning in Phase F. The detail of the Implementation and Migration Plan, Version 0.1 must be aligned to the detail of the Architecture Roadmap and be sufficient to identify the necessary projects and resource requirements to realize the roadmap.

When creating the Implementation and Migration Plan there are many approaches to consider, such as a data-driven sequence, where application systems that create data are implemented first, then applications that process the data. A clear understanding of the dependencies and lifecycle of in-place SBBs is required for an effective Implementation and Migration Plan.

Finally, update the Architecture Vision, Architecture Definition Document, and Architecture Requirements Specification with any additional relevant outcomes from this phase.

12.4 Outputs

The outputs of Phase E may include, but are not restricted to:

- Refined and updated version of the Architecture Vision phase deliverables, where applicable, including:
  - Architecture Vision, including definition of types and degrees of interoperability
  - Statement of Architecture Work (see Part IV, Section 32.2.20), updated if necessary
- Draft Architecture Definition Document (see Part IV, Section 32.2.3), including:
  - Baseline Business Architecture, Version 1.0 updated if necessary
  - Target Business Architecture, Version 1.0 updated if necessary
  - Baseline Data Architecture, Version 1.0 updated if necessary
  - Target Data Architecture, Version 1.0 updated if necessary
  - Baseline Application Architecture, Version 1.0 updated if necessary
  - Target Application Architecture, Version 1.0 updated if necessary
  - Baseline Technology Architecture, Version 1.0 updated if necessary
  - Target Technology Architecture, Version 1.0 updated if necessary
  - Transition Architecture, number and scope as necessary
  - Views corresponding to the selected viewpoints addressing key stakeholder concerns
- Draft Architecture Requirements Specification (see Part IV, Section 32.2.6), including:
  - Consolidated Gaps, Solutions, and Dependencies Assessment
- Capability Assessments, including:
  - Business Capability Assessment
  - IT Capability Assessment
- Architecture Roadmap (see Part IV, Section 32.2.7), including:
  - Work package portfolio:
    - Work package description (name, description, objectives)
    - Functional requirements
    - Dependencies
    - Relationship to opportunity
    - Relationship to Architecture Definition Document and Architecture Requirements Specification
    - Relationship to any capability increments
    - Business value
    - Implementation Factor Assessment and Deduction Matrix
    - Impact

  - Identification of Transition Architectures, if any, including:
    - Relationship to Architecture Definition Document
  - Implementation recommendations:
    - Criteria measures of effectiveness
    - Risks and issues
    - Solution Building Blocks (SBBs)
- Implementation and Migration Plan, Version 0.1, including:
  - Implementation and Migration Strategy

The outputs may include some or all of the following:

- Diagrams:
  - Project Context diagram
  - Benefits diagram

12.5 Approach

Phase E concentrates on how to deliver the architecture. It takes into account the complete set of gaps between the Target and Baseline Architectures in all architecture domains, and logically groups changes into work packages within the enterprise's portfolios. This is an effort to build a best-fit roadmap that is based upon the stakeholder requirements, the enterprise's business transformation readiness, identified opportunities and solutions, and identified implementation constraints. The key is to focus on the final target while realizing incremental business value.

Phase E is the initial step on the creation of the Implementation and Migration Plan which is completed in Phase F. It provides the basis of a well considered Implementation and Migration Plan that is integrated into the enterprise's portfolio in Phase F.

The following four concepts are key to transitioning from developing to delivering a Target Architecture:

- Architecture Roadmap
- Work Packages
- Transition Architectures
- Implementation and Migration Plan

The Architecture Roadmap lists individual work packages in a timeline that will realize the Target Architecture.

Each work package identifies a logical group of changes necessary to realize the Target Architecture.

A Transition Architecture describes the enterprise at an architecturally significant state between the Baseline and Target Architectures. Transition Architectures provide interim Target Architectures upon which the organization can converge.

The Implementation and Migration Plan provides a schedule of the projects that will realize the Target Architecture.

Chapter 13

Phase F: Migration Planning

This chapter addresses migration planning; that is, how to move from the Baseline to the Target Architectures by finalizing a detailed Implementation and Migration Plan.

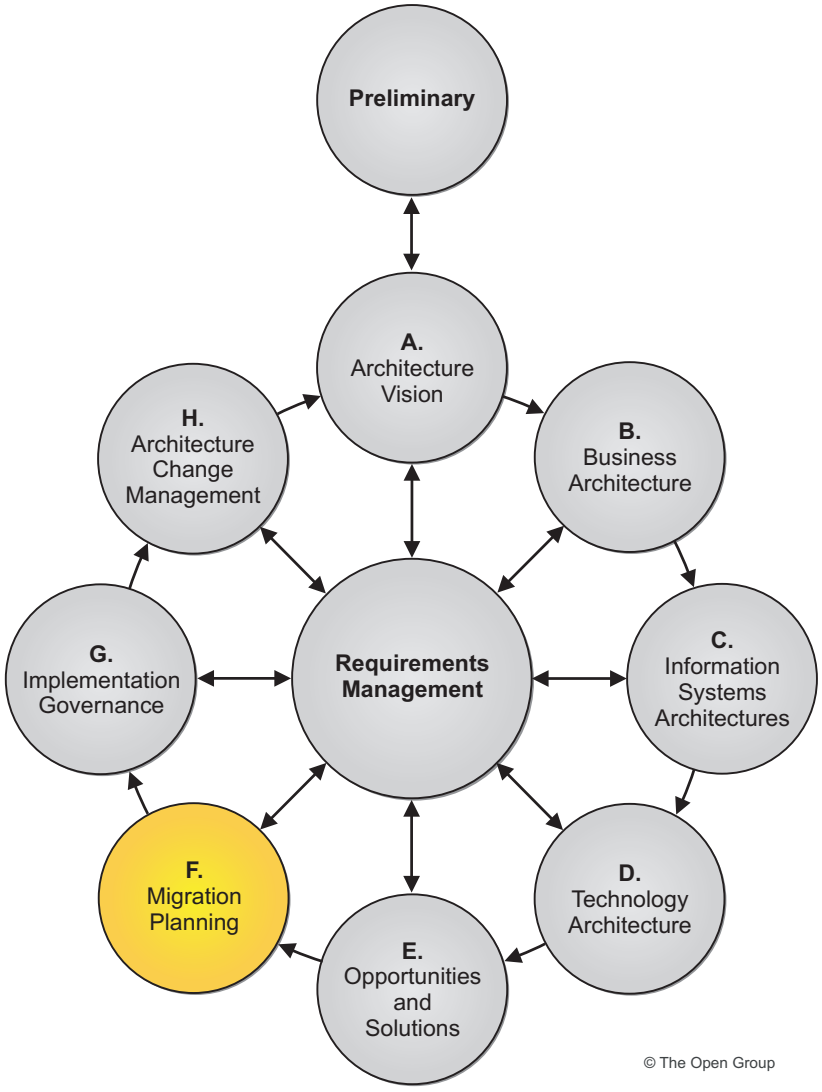

Figure 13-1 Phase F: Migration Planning

13.1 Objectives

The objectives of Phase F are to:

- Finalize the Architecture Roadmap and the supporting Implementation and Migration Plan
- Ensure that the Implementation and Migration Plan is co-ordinated with the enterprise's approach to managing and implementing change in the enterprise's overall change portfolio
- Ensure that the business value and cost of work packages and Transition Architectures is understood by key stakeholders

13.2 Inputs

This section defines the inputs to Phase F.

13.2.1 Reference Materials External to the Enterprise

- Architecture reference materials (see Part IV, Section 32.2.5)

13.2.2 Non-Architectural Inputs

- Request for Architecture Work (see Part IV, Section 32.2.17)
- Capability Assessment (see Part IV, Section 32.2.10)
- Communications Plan (see Part IV, Section 32.2.12)

13.2.3 Architectural Inputs

- Organizational Model for Enterprise Architecture (see Part IV, Section 32.2.16), including:
  - 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
- Governance models and frameworks for:
  - Corporate Business Planning
  - Enterprise Architecture
  - Portfolio, Program, Project Management

  - — System Development/Engineering
  - — Operations (Service)
- Tailored Architecture Framework (see Part IV, Section 32.2.21), including:
  - — Tailored architecture method
  - — Tailored architecture content (deliverables and artifacts)
  - — Configured and deployed tools
- Statement of Architecture Work (see Part IV, Section 32.2.20)
- Architecture Vision (see Part IV, Section 32.2.8)
- Architecture Repository (see Part IV, Section 32.2.5), including:
  - — Re-usable building blocks
  - — Publicly available reference models
  - — Organization-specific reference models
  - — Organization standards
- Draft Architecture Definition Document (see Part IV, Section 32.2.3), including:
  - — Baseline Business Architecture, Version 1.0 (detailed)
  - — Target Business Architecture, Version 1.0 (detailed)
  - — Baseline Data Architecture, Version 1.0 (detailed)
  - — Target Data Architecture, Version 1.0 (detailed)
  - — Baseline Application Architecture, Version 1.0 (detailed)
  - — Target Application Architecture, Version 1.0 (detailed)
  - — Baseline Technology Architecture, Version 1.0 (detailed)
  - — Target Technology Architecture, Version 1.0 (detailed)
  - — Transition Architectures, if any
- Draft Architecture Requirements Specification (see Part IV, Section 32.2.6), including:
  - — Architectural requirements
  - — Gap analysis results (from Business, Data, Application, and Technology Architecture)
  - — IT Service Management requirements
- Change Requests for existing business programs and projects (see Part IV, Section 32.2.11)
- Architecture Roadmap, Version 0.1 (see Part IV, Section 32.2.7), including:
  - — Identification of work packages
  - — Identification of Transition Architectures
  - — Implementation Factor Assessment and Deduction Matrix

- Capability Assessment (see Part IV, Section 32.2.10), including:
  - Business Capability Assessment
  - IT Capability Assessment
- Implementation and Migration Plan, Version 0.1 (see Part IV, Section 32.2.14) including the high-level Implementation and Migration Strategy

13.3 Steps

The level of detail addressed in Phase F will depend on the scope and goals of the overall architecture effort.

The order of the steps in Phase F as well as the time at which they are formally started and completed should be adapted to the situation at hand in accordance with the established Architecture Governance.

All activities that have been initiated in these steps must be closed during the "Complete the architecture development cycle and document lessons learned step" (see Section 13.3.7).

The steps in Phase F are as follows:

- Confirm management framework interactions for Implementation and Migration Plan (see Section 13.3.1)
- Assign a business value to each work package (see Section 13.3.2)
- Estimate resource requirements, project timings, and availability/delivery vehicle (see Section 13.3.3)
- Prioritize the migration projects through the conduct of a cost/benefit assessment and risk validation (see Section 13.3.4)
- Confirm Architecture Roadmap and update Architecture Definition Document (see Section 13.3.5)
- Complete the Implementation and Migration Plan (see Section 13.3.6)
- Complete the architecture development cycle and document lessons learned (see Section 13.3.7)

13.3.1 Confirm Management Framework Interactions for the Implementation and Migration Plan

This step is about co-ordinating the Implementation and Migration Plan with the management frameworks within the organization. There are typically four management frameworks that have to work closely together for the Implementation and Migration Plan to succeed:

- **Business Planning** that conceives, directs, and provides the resources for all of the activities required to achieve concrete business objectives/outcomes
- **Enterprise Architecture** that structures and gives context to all enterprise activities delivering concrete business outcomes primarily but not exclusively in the IT domain
- **Project/Portfolio Management** that co-ordinates, designs, and builds the business systems that deliver the concrete business outcomes

- **Operations Management** that integrates, operates, and maintains the deliverables that deliver the concrete business outcomes

The Implementation and Migration Plan will impact the outputs of each of these frameworks and consequently has to be reflected in them. In the course of this step, understand the frameworks within the organization and ensure that these plans are co-ordinated and inserted (in a summary format) within the plans of each one of these frameworks.

The outcome of this step may well be that the Implementation and Migration Plan could be part of a different plan produced by another one of the frameworks with Enterprise Architecture participation.

13.3.2 Assign a Business Value to Each Work Package

Establish and assign a business value to each of the work packages. The intent is to first establish what constitutes business value within the organization, how value can be measured, and then apply this to each one of the projects and project increments.

If Capability-Based Planning has been used, then the business values associated with the capabilities and associated capability increments should be used to assign the business values for deliverables.

There are several issues to address in this activity:

- **Performance Evaluation Criteria** are used by portfolio and capability managers to approve and monitor the progress of the architecture transformation
- **Return-on-Investment Criteria** have to be detailed and signed off by the various executive stakeholders
- **Business Value** has to be defined as well as techniques, such as the value chain, which are to be used to illustrate the role in achieving tangible business outcomes

  Business value will be used by portfolio and capability managers to allocate resources and, in cases where there are cutbacks, business value in conjunction with return on investment can be used to determine whether an endeavor proceeds, is delayed, or is canceled.
- **Critical Success Factors (CSFs)** should be established to define success for a project and/or project increment

  These will provide managers and implementers with a gauge as to what constitutes a successful implementation.
- **Measures of Effectiveness (MOE)** are often performance criteria and many corporations include them in the CSFs

  Where they are treated discretely, it should be clear as to how these criteria are to be grouped.
- **Strategic Fit** based upon the overall Enterprise Architecture (all tiers) will be the critical factor for allowing the approval of any new project or initiative and for determining the value of any deliverable

Use the work packages as a basis of identifying projects that will be in the Implementation and Migration Plan. The identified projects will be fully developed in other steps in Phase F. The projects, and project increments, may require adjustment of the Architecture Roadmap and Architecture Definition Document.

Risks should then be assigned to the projects and project increments by aggregating risks identified in the Consolidated Gaps, Solutions, and Dependencies Matrix (from Phase E).

Estimate the business value for each project using the Business Value Assessment Technique (see Part III, Section 24.5).

13.3.3 Estimate Resource Requirements, Project Timings, and Availability/Delivery Vehicle

This step determines the required resources and times for each project and their increments and provides the initial cost estimates. The costs should be broken down into capital (to create the capability) and operations and maintenance (to run and sustain the capability). Opportunities should be identified where the costs associated with delivering new and/or better capability can be offset by decommissioning existing systems. Assign required resources to each activity and aggregate them at the project increment and project level.

13.3.4 Prioritize the Migration Projects through the Conduct of a Cost/Benefit Assessment and Risk Validation

Prioritize the projects by ascertaining their business value against the cost of delivering them. The approach is to first determine, as clearly as possible, the net benefit of all of the SBBs delivered by the projects, and then verify that the risks have been effectively mitigated and factored in. Afterwards, the intent is to gain the requisite consensus to create a prioritized list of projects that will provide the basis for resource allocation.

It is important to discover all costs, and to ensure that decision-makers understand the net benefit over time.

Review the risks to ensure that the risks for the project deliverables have been mitigated as much as possible. The project list is then updated with risk-related comments.

Have the stakeholders agree upon a prioritization of the projects. Prioritization criteria will use elements identified in creation of the draft Architecture Roadmap in Phase E as well as those relating to individual stakeholders' agendas. Notice that it is possible for a project to earn a high priority if it provides a critical deliverable on the path to some large benefit, even if the immediate benefit of the project itself is small.

Formally review the risk assessment and revise it as necessary ensuring that there is a full understanding of the residual risk associated with the prioritization and the projected funding line.

13.3.5 Confirm Architecture Roadmap and Update Architecture Definition Document

Update the Architecture Roadmap including any Transition Architectures. Review the work to date to assess what the time-spans between Transition Architecture should be, taking into consideration the increments in business value and capability and other factors, such as risk. Once the capability increments have been finalized, consolidate the deliverables by project. This will result in a revised Architecture Roadmap.

This is needed in order to co-ordinate the development of several concurrent instances of the various architectures. A Transition Architecture State Evolution Table (see Part III, Section 24.4) can be used to show the proposed state of the domain architectures at various levels of detail.

If the implementation approach has shifted as a result of confirming the implementation increments, update the Architecture Definition Document. This may include assigning project objectives and aligning projects and their deliverables with the Transition Architectures to create an Architecture Definition Increments Table (see Part III, Section 24.3).

13.3.6 Complete the Implementation and Migration Plan

Generate the completed Implementation and Migration Plan. Much of the detail for the plan has already been gathered and this step brings it all together using accepted planning and management techniques.

This should include integrating all of the projects and activities as well as dependencies and impact of change into a project plan. Any Transition Architectures will act as portfolio milestones.

All external dependencies should be captured and included, and the overall availability of resources assessed. Project plans may be included within the Implementation and Migration Plan.

13.3.7 Complete the Architecture Development Cycle and Document Lessons Learned

This step transitions governance from the development of the architecture to the realization of the architecture. If the maturity of the Architecture Capability warrants, an Implementation Governance Model may be produced (see Part IV, Section 32.2.15).

Lessons learned during the development of the architecture should be documented and captured by the appropriate governance process in Phase H as inputs to managing the Architecture Capability.

The detail of the Architecture Roadmap and the Implementation and Migration Plan should be expressed at a similar level of detail to the Architecture Definition Document developed in Phases B, C, and D. Where significant additional detail is required by the next phase the architecture is likely transitioning to a different level. Depending upon the level of the Target Architecture and Implementation and Migration Plan it may be necessary to iterate another ADM cycle at a lower level of detail. See Part III, Chapter 18 and Chapter 19 for techniques to manage iteration and different levels of detail.

13.4 Outputs

The outputs of Phase F may include, but are not restricted to:

- Implementation and Migration Plan, Version 1.0 (see Part IV, Section 32.2.14), including:
  - Implementation and Migration Strategy
  - Project and portfolio breakdown of the implementation:
    - Allocation of work packages to project and portfolio
    - Capabilities delivered by projects
    - Relationship to Target Architecture and any Transition Architectures
    - Milestones and timing
    - Work breakdown structure
  - Project charters (optional):
    - Related work packages
    - Business value

  - Risk, issues, assumptions, dependencies
  - Resource requirements and costs
  - Benefits of migration
  - Estimated costs of migration options
- Finalized Architecture Definition Document (see Part IV, Section 32.2.3), including:
  - Finalized Transition Architectures, if any
- Finalized Architecture Requirements Specification (see Part IV, Section 32.2.6)
- Finalized Architecture Roadmap (see Part IV, Section 32.2.7)
- Re-Usable Architecture Building Blocks (see Part IV, Section 32.2.1)
- Requests for Architecture Work (see Part IV, Section 32.2.17) for a new iteration of the ADM cycle (if any)
- Implementation Governance Model (if any) (see Part IV, Section 32.2.15)
- Change Requests for the Architecture Capability arising from lessons learned

13.5 Approach

The focus of Phase F is the creation of an Implementation and Migration Plan in co-operation with the project and portfolio managers.

Phase E provides an incomplete Architecture Roadmap and Implementation and Migration Plan that address the Statement of Architecture Work. In Phase F this Roadmap and the Implementation and Migration Plan are integrated with the enterprise's other change activity.

Activities include assessing the dependencies, costs, and benefits of the various migration projects within the context of the enterprise's other activity. The Architecture Roadmap, Version 0.1 and Implementation and Migration Plan, Version 0.1 from Phase E will form the basis of the final Implementation and Migration Plan that will include portfolio and project-level detail.

The architecture development cycle should then be completed and lessons learned documented to enable continuous process improvement.

Chapter 14

Phase G: Implementation Governance

This chapter provides an architectural oversight of the implementation.

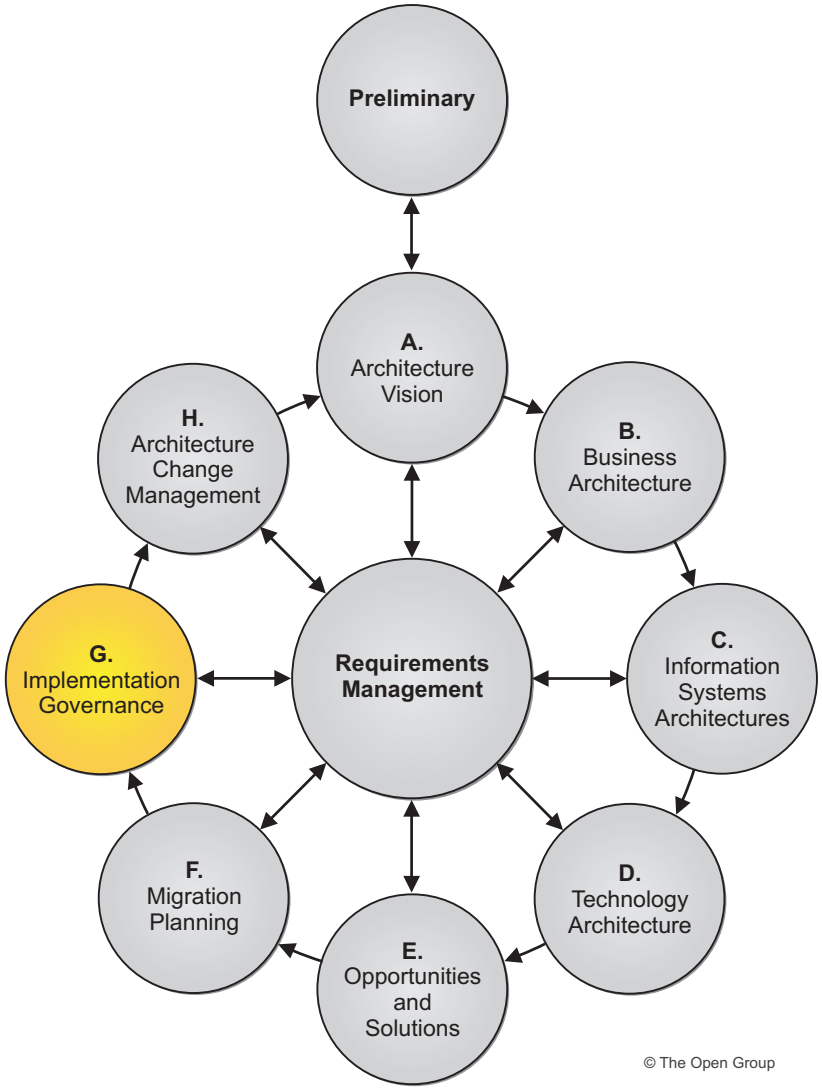

Figure 14-1 Phase G: Implementation Governance

14.1 Objectives

The objectives of Phase G are to:

- Ensure conformance with the Target Architecture by implementation projects
- Perform appropriate Architecture Governance functions for the solution and any implementation-driven architecture Change Requests

14.2 Inputs

This section defines the inputs to Phase G.

14.2.1 Reference Materials External to the Enterprise

- Architecture reference materials (see Part IV, Section 32.2.5)

14.2.2 Non-Architectural Inputs

- Request for Architecture Work (see Part IV, Section 32.2.17)
- Capability Assessment (see Part IV, Section 32.2.10)

14.2.3 Architectural Inputs

- Organizational Model for Enterprise Architecture (see Part IV, Section 32.2.16), including:
  - 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
- Tailored Architecture Framework (see Part IV, Section 32.2.21), including:
  - Tailored architecture method
  - Tailored architecture content (deliverables and artifacts)
  - Configured and deployed tools
- Statement of Architecture Work (see Part IV, Section 32.2.20)
- Architecture Vision (see Part IV, Section 32.2.8)
- Architecture Repository (see Part IV, Section 32.2.5), including:
  - Re-usable building blocks
  - Publicly available reference models

  - Organization-specific reference models
  - Organization standards
- Architecture Definition Document (see Part IV, Section 32.2.3)
- Architecture Requirements Specification (see Part IV, Section 32.2.6), including:
  - Architectural requirements
  - Gap analysis results (from Business, Data, Application, and Technology Architectures)
- Architecture Roadmap (see Part IV, Section 32.2.7)
- Implementation Governance Model (see Part IV, Section 32.2.15)
- Architecture Contract (standard) (see Part VI, Chapter 43)
- Request for Architecture Work (see Part IV, Section 32.2.17) identified during Phases E and F
- Implementation and Migration Plan (see Part IV, Section 32.2.14)

14.3 Steps

The level of detail addressed in Phase G will depend on the scope and goals of the overall architecture effort.

The order of the steps in Phase G as well as the time at which they are formally started and completed should be adapted to the situation at hand in accordance with the established Architecture Governance.

The steps in Phase G are as follows:

- Confirm scope and priorities for deployment with development management (see Section 14.3.1)
- Identify deployment resources and skills (see Section 14.3.2)
- Guide development of solutions deployment (see Section 14.3.3)
- Perform Enterprise Architecture Compliance reviews (see Section 14.3.4)
- Implement business and IT operations (see Section 14.3.5)
- Perform post-implementation review and close the implementation (see Section 14.3.6)

14.3.1 Confirm Scope and Priorities for Deployment with Development Management

- Review migration planning outputs and produce recommendations on deployment
- Identify Enterprise Architecture priorities for development teams
- Identify deployment issues and make recommendations
- Identify building blocks for replacement, update, etc.
- Perform gap analysis on Enterprise Architecture and solutions framework

  The gaps in the existing enterprise solutions framework need to be identified and the specific Solution Building Blocks (SBBs) required to fill these gaps will be identified by the Solution Architects. These SBBs may have a one-to-one or many-to-one relationship with

the projects. The Solution Architects need to define exactly how this will be done. There may be other projects working on these same capabilities and the Solution Architects need to ensure that they can leverage best value from these investments.

- Produce a gap analysis report

14.3.2 Identify Deployment Resources and Skills

The project resources will include the development resources which will need to be educated in the overall Enterprise Architecture deliverables and expectations from the specific development and implementation projects.

The following considerations should be addressed in this step:

- Identify system development methods required for solutions development

  Note: There are a range of systems development methods and tools available to the project teams. The method should ideally be able to interoperate with the architecture outputs; for example, generate code from architecture artifacts delivered to date. This could be achieved through the use of modeling languages used for the Enterprise Architecture development that may be captured as inputs to the systems development tools and thereby reduce the cost of solutions development.

- Ensure that the systems development method enables feedback to the architecture team on designs

14.3.3 Guide Development of Solutions Deployment

- Formulate project recommendation

  For each separate implementation and deployment project, do the following:

  — Document scope of individual project in impact analysis

  — Document strategic requirements (from the architectural perspective) in impact analysis

  — Document Change Requests (such as support for a standard interface) in impact analysis

  — Document rules for conformance in impact analysis

  — Document timeline requirements from roadmap in impact analysis

- Document Architecture Contract

  — Obtain signature from all developing organizations and sponsoring organization

- Update Enterprise Continuum directory and repository for solutions
- Guide development of business & IT operating models for services
- Provide service requirements derived from Enterprise Architecture
- Guide definition of business & IT operational requirements
- Carry out gap analysis between the Solution Architecture and operations
- Produce Implementation Plan

14.3.4 Perform Enterprise Architecture Compliance Reviews

- Review ongoing implementation governance and Architecture Compliance for each building block
- Conduct post-development reviews
- Close development part of deployment projects

14.3.5 Implement Business and IT Operations

- Carry out the deployment projects including: IT services delivery implementation; business services delivery implementation; skills development & training implementation; communications documentation publication
- Publish new Baseline Architectures to the Architecture Repository and update other impacted repositories, such as operational configuration management stores

14.3.6 Perform Post-Implementation Review and Close the Implementation

- Conduct post-implementation reviews
- Publish reviews and close projects

Closure on Phase G will be when the solutions are fully deployed once.

14.4 Outputs

The outputs of Phase G may include, but are not restricted to:

- Architecture Contract (signed) (see Part VI, Chapter 43), as recommended in the architecture-compliant implemented architectures
- Compliance Assessments (see Part IV, Section 32.2.13)
- Change Requests (see Part IV, Section 32.2.11)
- Architecture-compliant solutions deployed including:
  - The architecture-compliant implemented system

    Note: The implemented system is actually an output of the development process. However, given the importance of this output, it is stated here as an output of the ADM. The direct involvement of architecture staff in implementation will vary according to organizational policy, as described in Part VI, Chapter 44.
  - Populated Architecture Repository
  - Architecture compliance recommendations and dispensations
  - Recommendations on service delivery requirements
  - Recommendations on performance metrics
  - Service-Level Agreements (SLAs)
  - Architecture Vision, updated post-implementation

— Architecture Definition Document, updated post-implementation

— Business and IT operating models for the implemented solution

14.5 Approach

It is here that all the information for successful management of the various implementation projects is brought together. Note that, in parallel with Phase G, there is the execution of an organizational-specific development process, where the actual development happens.

To enable early realization of business value and benefits, and to minimize the risk in the transformation and migration program, the favored approach is to deploy the Target Architecture as a series of transitions. Each transition represents an incremental step towards the target, and each delivers business benefit in its own right. Therefore, the overall approach in Phase G is to:

- Establish an implementation program that will enable the delivery of the Transition Architectures agreed for implementation during the Migration Planning phase
- Adopt a phased deployment schedule that reflects the business priorities embodied in the Architecture Roadmap
- Follow the organization's standard for corporate, IT, and Architecture Governance
- Use the organization's established portfolio/program management approach, where this exists
- Define an operations framework to ensure the effective long life of the deployed solution

Phase G establishes the connection between architecture and implementation organization, through the Architecture Contract.

Project details are developed, including:

- Name, description, and objectives
- Scope, deliverables, and constraints
- Measures of effectiveness
- Acceptance criteria
- Risks and issues

Implementation governance is closely allied to overall Architecture Governance, which is discussed in Part VI, Chapter 44.

A key aspect of Phase G is ensuring compliance with the defined architecture(s), not only by the implementation projects, but also by other ongoing projects within the enterprise. The considerations involved with this are explained in detail in Part VI, Chapter 42.

Chapter 15

Phase H: Architecture Change Management

This chapter looks at establishing procedures for managing change to the new architecture.

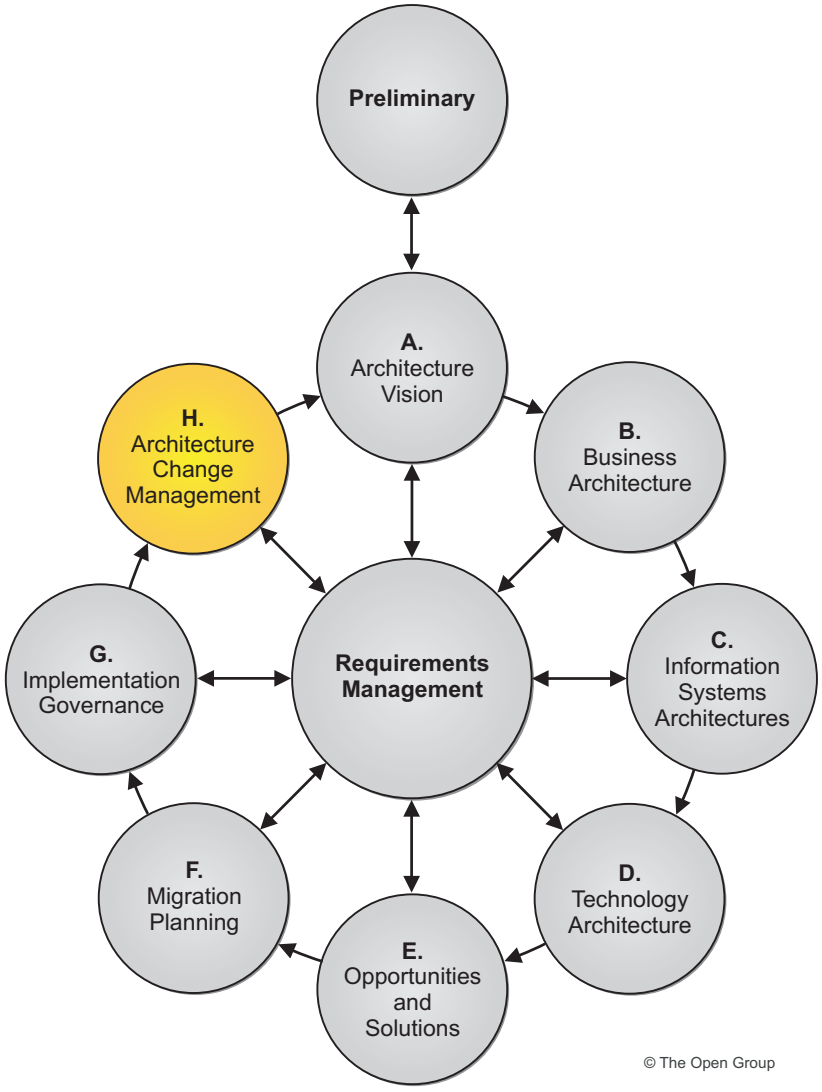

Figure 15-1 Phase H: Architecture Change Management

15.1 Objectives

The objectives of Phase H are to:

- Ensure that the architecture lifecycle is maintained
- Ensure that the Architecture Governance Framework is executed
- Ensure that the Enterprise Architecture Capability meets current requirements

15.2 Inputs

This section defines the inputs to Phase H.

15.2.1 Reference Materials External to the Enterprise

- Architecture reference materials (see Part IV, Section 32.2.5)

15.2.2 Non-Architectural Inputs

- Request for Architecture Work (see Part IV, Section 32.2.17)

15.2.3 Architectural Inputs

- Organizational Model for Enterprise Architecture (see Part IV, Section 32.2.16), including:
  - 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
- Tailored Architecture Framework (see Part IV, Section 32.2.21), including:
  - Tailored architecture method
  - Tailored architecture content (deliverables and artifacts)
  - Configured and deployed tools
- Statement of Architecture Work (see Part IV, Section 32.2.20)
- Architecture Vision (see Part IV, Section 32.2.8)
- Architecture Repository (see Part IV, Section 32.2.5), including:
  - Re-usable building blocks
  - Publicly available reference models

  - — Organization-specific reference models
  - — Organization standards
- Architecture Definition Document (see Part IV, Section 32.2.3)
- Architecture Requirements Specification (see Part IV, Section 32.2.6), including:
  - — Gap analysis results (from Business, Data, Application, and Technology Architectures)
  - — Architectural requirements
- Architecture Roadmap (see Part IV, Section 32.2.7)
- Change Request (see Part IV, Section 32.2.11), — technology changes:
  - — New technology reports
  - — Asset management cost reduction initiatives
  - — Technology withdrawal reports
  - — Standards initiatives
- Change Request (see Part IV, Section 32.2.11), — business changes:
  - — Business developments
  - — Business exceptions
  - — Business innovations
  - — Business technology innovations
  - — Strategic change developments
- Change Request (see Part IV, Section 32.2.11), — from lessons learned
- Implementation Governance Model (see Part IV, Section 32.2.15)
- Architecture Contract (signed) (see Part VI, Chapter 43)
- Compliance Assessments (see Part IV, Section 32.2.13)
- Implementation and Migration Plan (see Part IV, Section 32.2.14)

15.3 Steps

The level of detail addressed in Phase H will depend on the scope and goals of the overall architecture effort.

The order of the steps in Phase H as well as the time at which they are formally started and completed should be adapted to the situation at hand in accordance with the established Architecture Governance.

The steps in Phase H are as follows:

- Establish value realization process (see Section 15.3.1)
- Deploy monitoring tools (see Section 15.3.2)
- Manage risks (see Section 15.3.3)

- Provide analysis for architecture change management (see Section 15.3.4)
- Develop change requirements to meet performance targets (see Section 15.3.5)
- Manage governance process (see Section 15.3.6)
- Activate the process to implement change (see Section 15.3.7)

15.3.1 Establish Value Realization Process

Influence business projects to exploit the Enterprise Architecture for value realization (outcomes).

15.3.2 Deploy Monitoring Tools

Ensure monitoring tools are deployed and applied to enable the following:

- Monitor technology changes which could impact the Baseline Architecture
- Monitor business changes which could impact the Baseline Architecture
- Business value tracking; e.g., investment appraisal method to determine value metrics for the business objectives
- Monitor Enterprise Architecture Capability maturity
- Track and assess asset management programs
- Track the QoS performances and usage
- Determine and track business continuity requirements

15.3.3 Manage Risks

Manage Enterprise Architecture risks and provide recommendations for IT strategy.

15.3.4 Provide Analysis for Architecture Change Management

Provide analysis for architecture change management:

- Analyze performance
- Conduct Enterprise Architecture performance reviews with service management
- Assess Change Requests and reporting to ensure that the expected value realization and Service-Level Agreement (SLA) expectations of the customers are met
- Undertake a gap analysis of the performance of the Enterprise Architecture
- Ensure change management requests adhere to the Enterprise Architecture Governance and framework

15.3.5 Develop Change Requirements to Meet Performance Targets

Make recommendations on change requirements to meet performance targets and development of position to act.

15.3.6 Manage Governance Process

Manage governance process and framework for architecture:

- Arrange meeting of Architecture Board (or other Governing Council)
- Hold meeting of the Architecture Board with the aim of the meeting to decide on handling changes (technology and business and dispensations)

15.3.7 Activate the Process to Implement Change

Activate the architecture process to implement change:

- Produce a new Request for Architecture Work and request for investment
- Ensure any changes implemented in this phase are captured and documented in the Architecture Repository

15.4 Outputs

The outputs of Phase H may include, but are not restricted to:

- Architecture updates (for maintenance changes)
- Changes to architecture framework and principles (for maintenance changes)
- New Request for Architecture Work (see Part IV, Section 32.2.17), to move to another cycle (for major changes)
- Statement of Architecture Work (see Part IV, Section 32.2.20), updated if necessary
- Architecture Contract (see Part IV, Chapter 43), updated if necessary
- Compliance Assessments (see Part IV, Section 32.2.13), updated if necessary

15.5 Approach

The goal of an architecture change management process is to ensure that the architecture achieves its original target business value. This includes managing changes to the architecture in a cohesive and architected way.

This process will typically provide for the continual monitoring of such things as governance requests, new developments in technology, and changes in the business environment. When changes are identified, change management will determine whether to formally initiate a new architecture evolution cycle.

Additionally, the architecture change management process aims to establish and support the implemented Enterprise Architecture as a dynamic architecture; that is, one having the flexibility to evolve rapidly in response to changes in the technology and business environment.

Monitoring business growth and decline is a critical aspect of this phase. Usage of the Enterprise

Architecture is the most important part of the architecture development cycle. All too often the business has been left with an Enterprise Architecture that works for the organization of yesterday but may not give back sufficient capability to meet the needs of the enterprise of today and tomorrow.

In many cases the architecture continues to fit, but the solutions underlying them may not, and some changes are required. The Enterprise Architect needs to be aware of these change requirements and considers this an essential part of constant renewal of the architecture.

Capacity measurement and recommendations for planning are a key aspect of this phase. While the architecture has been built to deliver a steady state Business Architecture with agreed capacity during the lifecycle of this Enterprise Architecture, the growth or decline in usage needs to be continually assessed to ensure that maximum business value is achieved.

For example, some Solution Architectures may not lend themselves to be scalable by a large factor — say 10 — or alternative solutions may be more economic when scaled up. While the architecture specifications may not change, the solutions or their operational context may change.

If the performance management and reporting has been built into the work products through previous phases, then this phase is about ensuring the effectiveness of these. If there needs to be additional monitoring or reporting, then this phase will handle the changes.

The value and change management process, once established, will determine:

- The circumstances under which the Enterprise Architecture, or parts of it, will be permitted to change after deployment, and the process by which that will happen
- The circumstances under which the architecture development cycle will be initiated again to develop a new architecture

The architecture change management process is very closely related to the Architecture Governance processes of the enterprise, and to the management of the Architecture Contract (see Part VI, Chapter 43) between the architecture function and the business users of the enterprise.

In Phase H it is critical that the governance body establish criteria to judge whether a Change Request warrants just an architecture update or whether it warrants starting a new cycle of the Architecture Development Method (ADM). It is especially important to avoid "creeping elegance", and the governance body must continue to look for changes that relate directly to business value.

An Architecture Compliance report should state whether the change is compliant to the current architecture. If it is non-compliant, an exemption may be granted with valid rationale. If the change has high impact on the architecture, then a strategy to manage its impact should be defined.

Guidelines for establishing these criteria are difficult to prescribe, as many companies accept risk differently, but as the ADM is exercised, the maturity level of the governance body will improve, and criteria will become clear for specific needs.

15.5.1 Drivers for Change

The main purpose for the development of the Enterprise Architecture so far has been strategic direction and top-down architecture and project generation to achieve corporate capabilities. However, Enterprise Architecture does not operate in a vacuum. There is usually an existing infrastructure and business which is already providing value.

There are also probably drivers for change which are often bottom-up, based upon modifying the existing infrastructure to enhance functionality. Enterprise Architecture changes this paradigm by a strategic top-down approach to a degree, although the delivery of increments makes the equation more complex.

There are three ways to change the existing infrastructure that have to be integrated:

- Strategic, top-down directed change to enhance or create new capability (capital)
- Bottom-up changes to correct or enhance capability (operations and maintenance) for infrastructure under operations management
- Experiences with the previously delivered project increments in the care of operations management, but still being delivered by ongoing projects

Governance will have to handle the co-ordination of these Requests for Change, plus there needs to be a lessons learned process to allow for problems with the recently delivered increments to be resolved and changes made to the Target Architectures being designed and planned.

A lessons learned process ensures that mistakes are made once and not repeated. They can come from anywhere and anyone and cover any aspect of the Enterprise Architecture at any level (strategic, Enterprise Architecture definition, transition, or project). Often an Enterprise Architecture-related lesson may be an indirect outcome of a lesson learned elsewhere in the organization.

The Architecture Board (see Part VI, Chapter 41) assesses and approves Requests for Change (RFC). An RFC is typically in response to known problems but can also include improvements. A challenge for the Architecture Board when handling an RFC is to determine whether it should be approved or whether a project in a Transition Architecture will resolve the issue.

When assessing project or solution fit into the architecture, there may also be the case when an innovative solution or RFC drives a change in the architecture.

In addition, there are many technology-related drivers for architecture Change Requests. For example:

- New technology reports
- Asset management cost reductions
- Technology withdrawal
- Standards initiatives

This type of Change Request is normally manageable primarily through an enterprise's change management and Architecture Governance processes.

In addition, there are business drivers for architecture change, including:

- Business-as-usual developments
- Business exceptions

- Business innovations
- Business technology innovations
- Strategic change

This type of Change Request often results in a complete re-development of the architecture, or at least in an iteration of a part of the architecture development cycle, as explained below.

15.5.2 Enterprise Architecture Change Management Process

The Enterprise Architecture change management process needs to determine how changes are to be managed, what techniques are to be applied, and what methodologies used. The process also needs a filtering function that determines which phases of the architecture development process are impacted by requirements. For example, changes that affect only migration may be of no interest in the architecture development phases.

There are many valid approaches to change management, and various management techniques and methodologies that can be used to manage change; for example, project management methods such as PRINCE2, service management methods such as ITIL, management consultancy methods such as Catalyst, and many others. An enterprise that already has a change management process in place in a field other than architecture (for example, in systems development or project management) may well be able to adapt it for use in relation to architecture.

The following describes an approach to change management, aimed particularly at the support of a dynamic Enterprise Architecture, which may be considered for use if no similar process currently exists.

The approach is based on classifying required architectural changes into one of three categories:

- **Simplification change**: a simplification change can normally be handled via change management techniques
- **Incremental change**: an incremental change may be capable of being handled via change management techniques, or it may require partial re-architecting, depending on the nature of the change (see Section 15.5.3 for guidelines)
- **Re-architecting change**: a re-architecting change requires putting the whole architecture through the architecture development cycle again

Another way of looking at these three choices is to say that a simplification change to an architecture is often driven by a requirement to reduce investment; an incremental change is driven by a requirement to derive additional value from existing investment; and a re-architecting change is driven by a requirement to increase investment in order to create new value for exploitation.

To determine whether a change is simplification, incremental, or re-architecting, the following activities are undertaken:

1. Registration of all events that may impact the architecture
2. Resource allocation and management for architecture tasks
3. The process or role responsible for architecture resources has to make assessment of what should be done
4. Evaluation of impacts

15.5.3 Guidelines for Maintenance versus Architecture Redesign

A good guideline is:

- If the change impacts two stakeholders or more, then it is likely to require an architecture redesign and re-entry to the ADM
- If the change impacts only one stakeholder, then it is more likely to be a candidate for change management
- If the change can be allowed under a dispensation, then it is more likely to be a candidate for change management

For example:

- If the impact is significant for the business strategy, then there may be a need to redo the whole Enterprise Architecture — thus a re-architecting approach
- If a new technology or standards emerge, then there may be a need to refresh the Technology Architecture, but not the whole Enterprise Architecture — thus an incremental change
- If the change is at an infrastructure level — for example, ten systems reduced or changed to one system — this may not change the architecture above the physical layer, but it will change the Baseline Description of the Technology Architecture; this would be a simplification change handled via change management techniques

In particular, a refreshment cycle (partial or complete re-architecting) may be required if:

- The Foundation Architecture needs to be re-aligned with the business strategy
- Substantial change is required to components and guidelines for use in deployment of the architecture
- Significant standards used in the product architecture are changed which have significant end-user impact; e.g., regulatory changes

If there is a need for a refreshment cycle, then a new Request for Architecture Work must be issued (to move to another cycle).

Chapter 16

ADM Architecture Requirements Management

This chapter looks at the process of managing architecture requirements throughout the ADM.

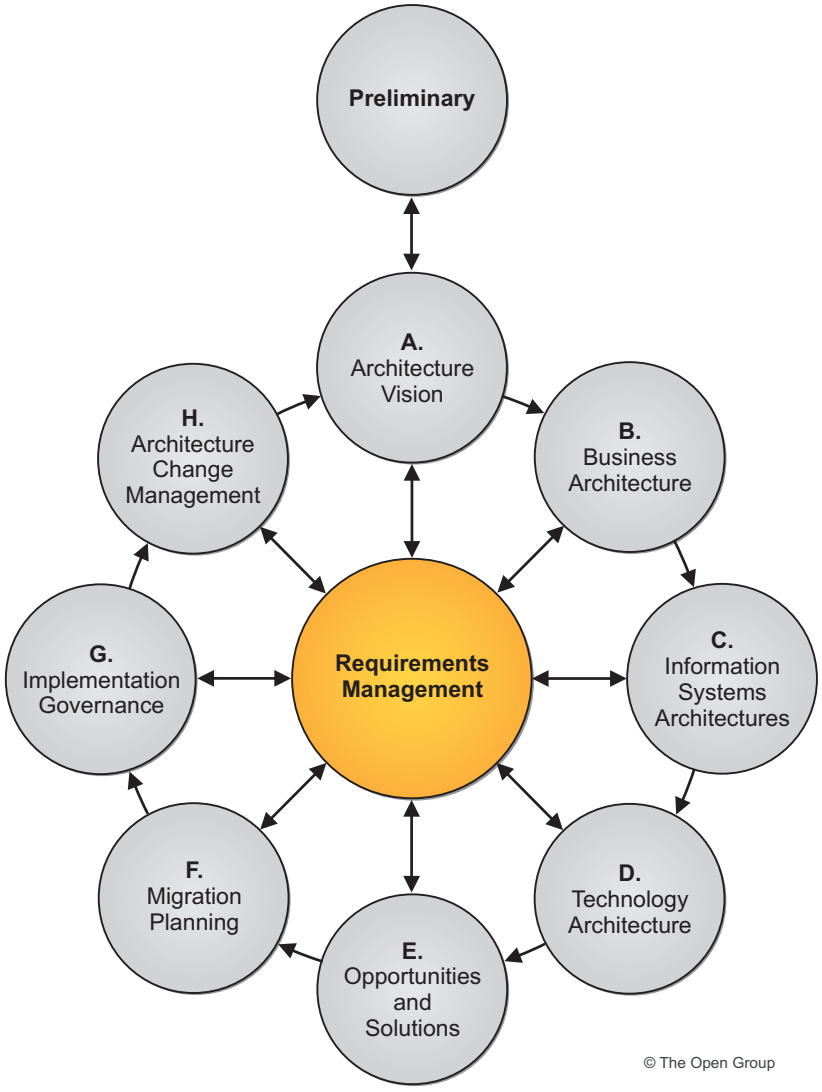

Figure 16-1 ADM Architecture Requirements Management

16.1 Objectives

The objectives of the Requirements Management phase are to:

- Ensure that the Requirements Management process is sustained and operates for all relevant ADM phases
- Manage architecture requirements identified during any execution of the ADM cycle or a phase
- Ensure that relevant architecture requirements are available for use by each phase as the phase is executed

16.2 Inputs

Inputs to the Requirements Management phase are:

- A populated Architecture Repository (see Part IV, Section 32.2.5)
- Organizational Model for Enterprise Architecture (see Part IV, Section 32.2.16), including:
  - 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
- Tailored Architecture Framework (see Part IV, Section 32.2.21), including:
  - Tailored architecture method
  - Tailored architecture content (deliverables and artifacts)
  - Configured and deployed tools
- Statement of Architecture Work (see Part IV, Section 32.2.20)
- Architecture Vision (see Part IV, Section 32.2.8)
- Architecture requirements, populating an Architecture Requirements Specification (see Part IV, Section 32.2.6)
- Requirements Impact Assessment (see Part IV, Section 32.2.18)

16.3 Steps

The steps in the Requirements Management phase are described in the table below:

	Requirements Management Steps	ADM Phase Steps
Step 1		Identify/document requirements — use business scenarios, or an analogous technique.

	Requirements Management Steps	ADM Phase Steps
Step 2	Baseline requirements: a. Determine priorities arising from current phase of ADM b. Confirm stakeholder buy-in to resultant priorities c. Record requirements priorities and place in Architecture Requirements Repository	
Step 3	Monitor baseline requirements.	
Step 4		Identify changed requirements: a. Remove or re-assess priorities b. Add requirements and re-assess priorities c. Modify existing requirements

	Requirements Management Steps	ADM Phase Steps
Step 5	Identify changed requirements and record priorities: a. Identify changed requirements and ensure the requirements are prioritized by the architect(s) responsible for the current phase, and by the relevant stakeholders b. Record new priorities c. Ensure that any conflicts are identified and managed through the phases to a successful conclusion and prioritization d. Generate Requirements Impact Statement (see Section 32.2.18) for steering the architecture team **Notes** ■ Changed requirements can come in through any route To ensure that the requirements are properly assessed and prioritized, this process needs to direct the ADM phases and record the decisions related to the requirements. ■ The Requirements Management phase needs to determine stakeholder satisfaction with the decisions Where there is dissatisfaction, the phase remains accountable to ensure the resolution of the issues and determine next steps.	

	Requirements Management Steps	ADM Phase Steps
Step 6		a. Assess impact of changed requirements on current (active) phase b. Assess impact of changed requirements on previous phases c. Determine whether to implement change, or defer to later ADM cycle; if decision is to implement, assess timescale for change management implementation d. Issue Requirements Impact Statement, Version *n*+1
Step 7		Implement requirements arising from Phase H. The architecture can be changed through its lifecycle by the Architecture Change Management phase (Phase H). The Requirements Management process ensures that new or changing requirements that are derived from Phase H are managed accordingly.
Step 8	Update the Architecture Requirements Repository with information relating to the changes requested, including stakeholder views affected.	
Step 9		Implement change in the current phase.

	Requirements Management Steps	ADM Phase Steps
Step 10		Assess and revise gap analysis for past phases. The gap analysis in the ADM Phases B through D identifies the gaps between Baseline and Target Architectures. Certain types of gap can give rise to gap requirements. The ADM describes two kinds of gap: ■ Something that is present in the baseline, but not in the target (i.e., eliminated — by accident or design) ■ Something not in the baseline, but present in the target (i.e., new) A "gap requirement" is anything that has been eliminated by accident, and therefore requires a change to the Target Architecture. If the gap analysis generates gap requirements, then this step will ensure that they are addressed, documented, and recorded in the Architecture Requirements Repository, and that the Target Architecture is revised accordingly.

16.4 Outputs

The outputs of the Requirements Management process may include, but are not restricted to:

- Requirements Impact Assessment (see Part IV, Section 32.2.18)
- Updated Architecture Requirements Specification (see Part IV, Section 32.2.6), if necessary

The Architecture Requirements Repository will be updated as part of the Requirements Management phase and should contain all requirements information.

When new requirements arise, or existing ones are changed, a Requirements Impact Statement is generated, which identifies the phases of the ADM that need to be revisited to address the changes. The statement goes through various iterations until the final version, which includes the full implications of the requirements (e.g., costs, timescales, and business metrics) on the architecture development. Once requirements for the current ADM cycle have been finalized then the Architecture Requirements Specification should be updated.

16.5 Approach

16.5.1 General

As indicated by the "Requirements Management" circle at the center of the ADM graphic, the ADM is continuously driven by the Requirements Management process.

It is important to note that the Requirements Management circle denotes not a static set of requirements, but a dynamic process whereby requirements for Enterprise Architecture and subsequent changes to those requirements are identified, stored, and fed into and out of the relevant ADM phases, and also between cycles of the ADM.

The ability to deal with changes in requirements is crucial. Architecture is an activity that by its very nature deals with uncertainty and change — the "grey area" between what stakeholders aspire to and what can be specified and engineered as a solution. Architecture requirements are therefore invariably subject to change in practice. Moreover, architecture often deals with drivers and constraints, many of which by their very nature are beyond the control of the enterprise (changing market conditions, new legislation, etc.), and which can produce changes in requirements in an unforeseen manner.

Note also that the Requirements Management process itself does not dispose of, address, or prioritize any requirements; this is done within the relevant phase of the ADM. It is merely the process for managing requirements throughout the overall ADM.

It is recommended that an Architecture Requirements Repository (see Part IV, Section 37.6) is used to record and manage all architecture requirements. Unlike the Architecture Requirements Specification, and the Requirements Impact Assessment, the Architecture Requirements Repository can hold information from multiple ADM cycles.

16.5.2 Requirements Development

The first high-level requirements are articulated as part of the Architecture Vision, generated by means of the business scenario or analogous technique.

Each phase of the ADM, from Preliminary to Phase H, must select the approved requirements for that phase as held in the Architecture Requirements Repository and Architecture Requirements Specification. At the completion of the phase the status of all such requirements needs to be updated. During the phase execution, new requirements generated for future architecture work within the scope of the current Statement of Architecture Work need to be documented within the Architecture Requirements Specification, and new requirements which are outside of the scope of the current Statement of Architecture Work must be input to the Architecture Requirements Repository for management through the Requirements Management process.

In each relevant phase of the ADM the architect should identify types of requirement that must be met by the architecture, including applicable:

- Functional requirements
- Non-functional requirements

When defining requirements the architect should take into account:

- Assumptions for requirements
- Constraints for requirements

- Domain-specific principles that drive requirements
- Policies affecting requirements
- Standards that requirements must meet
- Organization guidelines for requirements
- Specifications for requirements

Deliverables in later ADM phases also contain mappings to the design requirements, and may also generate new types of requirements (for example, conformance requirements, time windows for implementation).

16.5.3 Resources

The world of requirements engineering is rich with emerging recommendations and processes for Requirements Management. The TOGAF standard does not mandate or recommend any specific process or tool; it simply states what an effective Requirements Management process should achieve (i.e., the "requirements for requirements", if you like).

16.5.3.1 Business Scenarios

The business scenarios technique is an appropriate and effective technique to discover and document business requirements. Business scenarios are described in detail in the TOGAF® Series Guide: Business Scenarios.

16.5.3.2 Requirements Tools

There is a large, and increasing, number of Commercial Off-The-Shelf (COTS) tools available for the support of Requirements Management, albeit not necessarily designed for architecture requirements. The Volere website has a very useful list of leading requirements tools (see www.volere.co.uk/tools.htm).

The TOGAF Standard, Version 9.2

Part III:

ADM Guidelines and Techniques

The Open Group

Chapter 17

Introduction to Part III

This chapter provides an introduction to the guidelines and techniques provided in Part III: ADM Guidelines & Techniques.[2]

17.1 Guidelines for Adapting the ADM Process

The Architecture Development Method (ADM) process can be adapted to deal with a number of different usage scenarios, including different process styles (e.g., the use of iteration) and also specific specialist architectures (such as security). Guidelines included within this part are as follows:

- Applying Iteration to the ADM (see Chapter 18) discusses the concept of iteration and shows potential strategies for applying iterative concepts to the ADM
- Applying the ADM across the Architecture Landscape (see Chapter 19) discusses the different types of architecture engagement that may occur at different levels of the enterprise — this section then also discusses how the ADM process can be focused to support different types of engagement

17.2 Techniques for Architecture Development

The following techniques are described within Part III: ADM Guidelines & Techniques to support specific tasks within the ADM:

- Architecture Principles (see Chapter 20) — principles for the use and deployment of IT resources across the enterprise — describes how to develop the set of general rules and guidelines for the architecture being developed
- Stakeholder Management (see Chapter 21) describes stakeholder management, an important discipline that successful architecture practitioners can use to win support for their projects
- Architecture Patterns (see Chapter 22) provides guidance on using architectural patterns
- Gap Analysis (see Chapter 23) describes the technique known as gap analysis; it is widely used in the TOGAF ADM to validate an architecture that is being developed
- Migration Planning Techniques (see Chapter 24) describes a number of techniques to support migration planning in Phases E and F

2. Additional guidelines and techniques are available in the TOGAF Library.

- Interoperability Requirements (see Chapter 25) describes a technique for determining interoperability requirements
- Business Transformation Readiness Assessment (see Chapter 26) describes a technique for identifying business transformation issues
- Risk Management (see Chapter 27) describes a technique for managing risk during an architecture/business transformation project
- Capability-Based Planning (see Chapter 28) describes the technique of capability-based planning

17.3 Using the TOGAF Framework with Different Architectural Styles

The TOGAF framework is designed to be flexible and it can be used with various architectural styles. Further information can be found in the following Guides:

- Integrating Risk and Security within a TOGAF® Enterprise Architecture
- TOGAF® Series Guide: Using the TOGAF® Framework to Define and Govern Service-Oriented Architectures

Architectural styles differ in terms of focus, form, techniques, materials, subject, and time period. Some styles can be considered as fashionable, others focused on particular aspects of Enterprise Architecture. The TOGAF standard is a generic framework intended to be used in a wide variety of environments. It is a flexible and extensible framework that can be readily adapted to a number of architectural styles.

An organization's Architecture Landscape can be expected to contain architecture work that is developed in many architectural styles. The TOGAF standard ensures that the needs of each stakeholder are appropriately addressed in the context of other stakeholders and the Baseline Architecture.

When using the TOGAF standard to support a specific architectural style the practitioner must take into account the combination of distinctive features in which architecture is performed or expressed. As a first step, the distinctive features of a style must be identified.

For example, The Open Group definition for SOA identifies the following distinctive features:

- It is based on the design of the services — which mirror real-world business activities — comprising the enterprise (or inter-enterprise) business processes
- Service representation utilizes business descriptions to provide context (i.e., business process, goal, rule, policy, service interface, and service component) and implements services using service orchestration
- It places unique requirements on the infrastructure — it is recommended that implementations use open standards to realize interoperability and location transparency
- Implementations are environment-specific — they are constrained or enabled by context and must be described within that context

The second step is determining how these distinctive features will be addressed. Addressing a distinctive style should not call for significant changes to the TOGAF framework; instead it should adjust the models, viewpoints, and tools used by the practitioner.

In Phase B, Phase C, and Phase D the practitioner is expected to select the relevant architecture resources, including models, viewpoints, and tools, to properly describe the architecture domain and demonstrate that stakeholder concerns are addressed (see Part II, Section 7.3.1, Section 9.3.1, Section 10.3.1, and Section 11.3.1). Depending upon the distinctive features, different

architectural styles will add new elements that must be described, highlight existing elements, adjust the notation used to describe the architecture, and focus the architect on some stakeholders or stakeholder concerns.

Addressing the distinctive features will usually include extensions to the Architecture Content Metamodel and the use of specific notation or modeling techniques and the identification of viewpoints. Whether the style is dominant will determine whether it is necessary to revisit the Preliminary Phase and make changes to the Architecture Capability or whether support for the distinctive feature is possible within the scope of selection expected within a single ADM cycle.

Style-specific reference models and maturity models are commonly used tools that support a practitioner.

During the lifetime of the TOGAF framework many architectural styles have been developed to address key problems facing practitioners and to demonstrate how the TOGAF framework can be made more relevant within defined contexts.

Some of these have been developed by The Open Group Forums and Work Groups working in specific areas and have been published in Guides, White Papers, and Standards. Examples include:

- TOGAF® Series Guide: Using the TOGAF® Framework to Define and Govern Service-Oriented Architectures
- Integrating Risk and Security within a TOGAF® Enterprise Architecture

Some of these have been developed collaboratively between The Open Group and other bodies. Examples include:

- TOGAF® and SABSA® Integration
- Integrating the TOGAF® Standard with the BIAN Service Landscape
- Exploring Synergies between TOGAF® and Frameworx
- TOGAF® 9 and DoDAF 2.0

The TOGAF Library (see https://publications.opengroup.org/togaf-library) includes an evolving list of documents providing advice and guidance for the application of the TOGAF framework within specific contexts.

Chapter 18

Applying Iteration to the ADM

18.1 Overview

The graphical representation of the TOGAF ADM, as shown in Figure 4-1, and the description of the ADM phases discretely in order in Part II, can be read to imply a deterministic waterfall methodology. This method of presentation is provided for the purpose of quickly communicating the basics of architecture development and the architecture lifecycle. In practice, two key concepts are used to manage the complexity of developing an Enterprise Architecture and managing its lifecycle — iteration and levels (see Chapter 19). The two concepts are tightly linked.

The ADM supports a number of concepts that are characterized as iteration. First, iteration describes the process of both describing a comprehensive Architecture Landscape through multiple ADM cycles based upon individual initiatives bound to the scope of the Request for Architecture Work. Second, iteration describes the integrated process of developing an architecture where the activities described in different ADM phases interact to produce an integrated architecture. In order to concisely describe the activity and outputs, this latter iteration is described in sequential terms. Third, iteration describes the process of managing change to the organization's Architecture Capability.

Iteration to develop a comprehensive Architecture Landscape:

- Projects will exercise through the entire ADM cycle, commencing with Phase A

  Each cycle of the ADM will be bound by a Request for Architecture Work. The architecture output will populate the Architecture Landscape, either extending the landscape described, or changing the landscape where required.

- Separate projects may operate their own ADM cycles concurrently, with relationships between the different projects

- One project may trigger the initiation of another project

  Typically, this is used when higher-level architecture initiatives identify opportunities or solutions that require more detailed architecture, or when a project identifies landscape impacts outside the scope of its Request for Architecture Work.

Iteration within an ADM cycle (Architecture Development iteration):

- Projects may operate multiple ADM phases concurrently

  Typically, this is used to manage the inter-relationship between Business Architecture, Information Systems Architecture, and Technology Architecture.

- Projects may cycle between ADM phases, in planned cycles covering multiple phases

  Typically, this is used to converge on a detailed Target Architecture when higher-level

architecture does not exist to provide context and constraint.

- Projects may return to previous phases in order to circle back and update work products with new information

  Typically, this is used to converge on an executable Architecture Roadmap or Implementation and Migration Plan, when the implementation details and scope of change trigger a change or re-prioritization of stakeholder requirements.

Iteration to manage the Architecture Capability (Architecture Capability iteration):

- Projects may require a new iteration of the Preliminary Phase to (re-)establish aspects of the Architecture Capability identified in Phase A to address a Request for Architecture Work
- Projects may require a new iteration of the Preliminary Phase to adjust the organization's Architecture Capability as a result of identifying new or changed requirements for Architecture Capability as a result of a Change Request in Phase H

18.2 Iteration Cycles

The suggested iteration cycles for the TOGAF ADM are shown in Figure 18-1, and can be used to effectively group related architectural activities to achieve a specific purpose. These iteration cycles are referenced in Section 18.3 and Section 18.5.

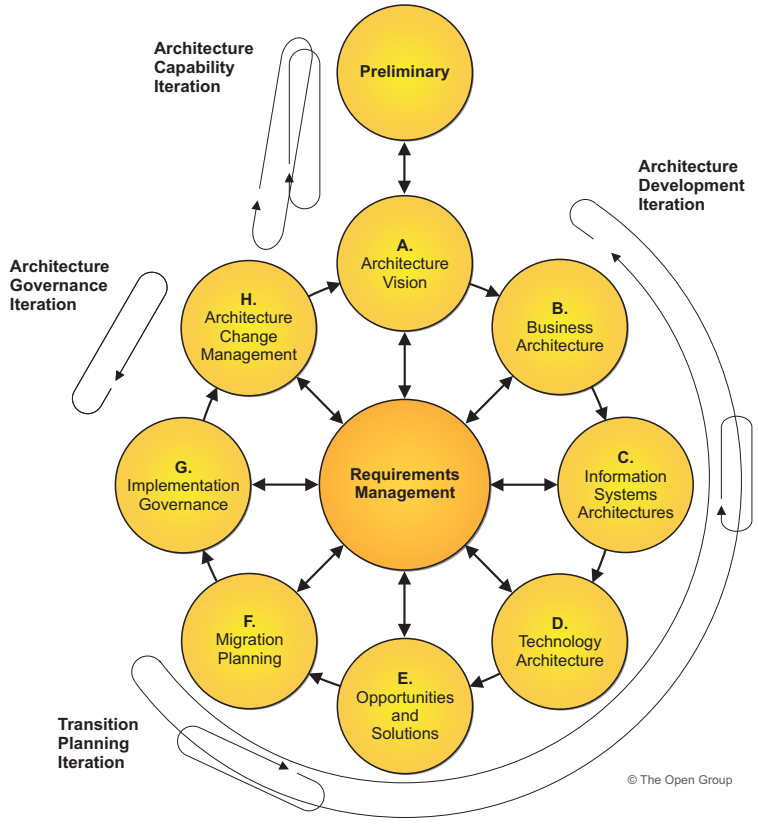

Figure 18-1 Iteration Cycles

- **Architecture Capability** iterations support the creation[3] and evolution of the required Architecture Capability

  This includes the initial mobilization of the architecture activity for a given purpose or architecture engagement type by establishing or adjusting the architecture approach, principles, scope, vision, and governance.

- **Architecture Development** iterations allow the creation of architecture content by cycling through, or integrating, Business, Information Systems, and Technology Architecture phases

  These iterations ensure that the architecture is considered as a whole. In this type of iteration stakeholder reviews are typically broader. As the iterations converge on a target, extensions into the Opportunities & Solutions and Migration Planning phases ensure that the architecture's implementability is considered as the architecture is finalized.

- **Transition Planning** iterations support the creation of formal change roadmaps for a defined architecture

- **Architecture Governance** iterations support governance of change activity progressing towards a defined Target Architecture

18.3 Classes of Architecture Engagement

An architecture function or services organization may be called upon to assist an enterprise in a number of different contexts, as the architectures developed can range from summary to detail, broad to narrow coverage, and current state to future state. In these contexts the concept of iteration should be used in developing the architecture.

Typically, there are three areas of engagement for architects:

- **Identification of Required Change**: outside the context of any change initiative, architecture can be used as a technique to provide visibility of the IT capability in order to support strategic decision-making and alignment of execution

- **Definition of Change**: where a need to change has been identified, architecture can be used as a technique to define the nature and extent of change in a structured fashion

  Within largescale change initiatives, architectures can be developed to provide detailed Architecture Definition for change initiatives that are bounded by the scope of a program or portfolio.

- **Implementation of Change**: architecture at all levels of the enterprise can be used as a technique to provide design governance to change initiatives by providing big-picture visibility, supplying structural constraints, and defining criteria on which to evaluate technical decisions

Figure 18-2 and the following table show the classes of Enterprise Architecture engagement.

3. Guidance on how to use a full ADM cycle for initially establishing an organization's Architecture Capability is found in Part VI, Chapter 40.

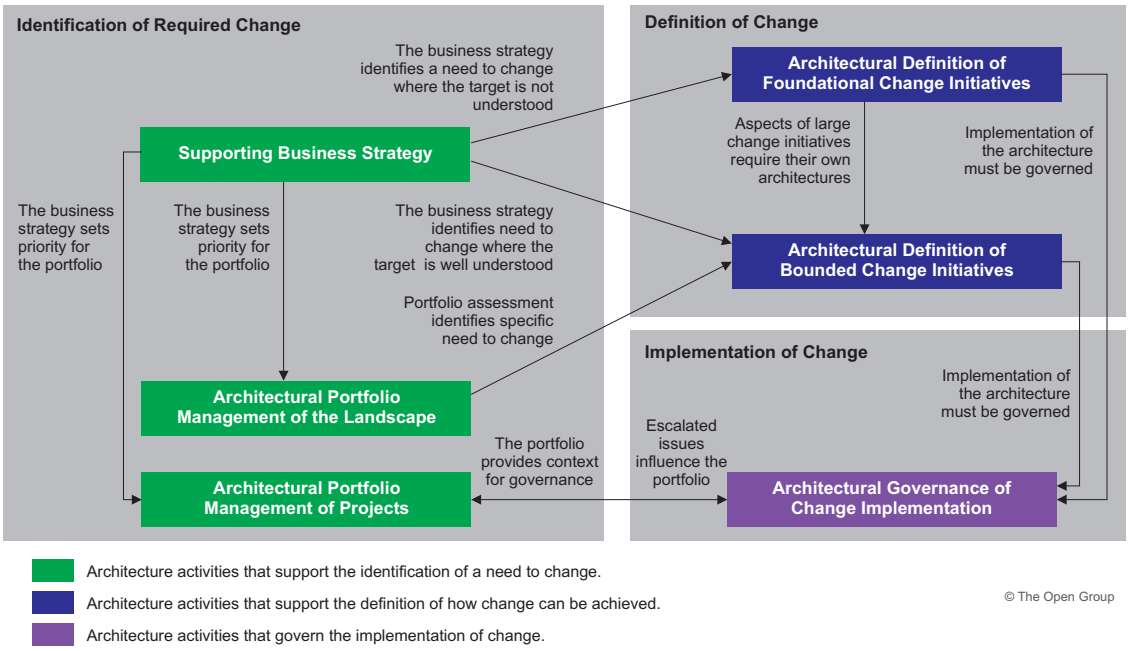

Figure 18-2 Classes of Enterprise Architecture Engagement

Each of these architecture engagement types is described in the table below.

Area of Engagement	Architecture Engagement	Description
Identification of Required Change	Supporting Business Strategy	As the business strategies, objectives, goals, and drivers change, it is necessary for the enterprise to change in order to maintain alignment. The creation of new business strategies can be supported by Enterprise Architecture by: ■ Providing visibility of change opportunities ■ Providing elaboration on the practical impacts of a particular strategic choice ■ Providing tests on the feasibility or viability of a particular strategic direction

Area of Engagement	Architecture Engagement	Description
	Architectural Portfolio Management of the Landscape	It is common practice across large organizations for a service management organization to provide operational reporting and management of the IT portfolio. Enterprise Architecture can add a further dimension to service management reporting, by supporting a linkage between operational performance and the strategic need for IT. Using the traceability between IT and business inherent in Enterprise Architecture, it is possible to evaluate the IT portfolio against operational performance data and business needs (e.g., cost, functionality, availability, responsiveness) to determine areas where misalignment is occurring and change needs to take place.
	Architectural Portfolio Management of Projects	It is common practice across large organizations for a program management organization to provide operational reporting and management of the change portfolio. Enterprise Architecture can add a further dimension to project portfolio management reporting, by supporting a linkage between project scope, architectural impact, and business value. Architectural factors can be added to other quantitative project factors to support strategic decision-making on project priority and funding levels.
Definition of Change	Architectural Definition of Foundational Change Initiatives	Foundational change initiatives are change efforts that have a known objective, but are not strictly scoped or bounded by a shared vision or requirements. In foundational change initiatives, the initial priority is to understand the nature of the problem and to bring structure to the definition of the problem. Once the problem is more effectively understood, it is possible to define appropriate solutions and to align stakeholders around a common vision and purpose.

Area of Engagement	Architecture Engagement	Description
	Architectural Definition of Bounded Change Initiatives	Bounded change initiatives are change efforts that typically arise as the outcome of a prior architectural strategy, evaluation, or vision. In bounded change initiatives, the desired outcome is already understood and agreed upon. The focus of architectural effort in this class of engagement is to effectively elaborate a baseline solution that addresses the identified requirements, issues, drivers, and constraints.
Implementation of Change	Architectural Governance of Change Implementation	Once an architectural solution model has been defined, it provides a basis for design and implementation. In order to ensure that the objectives and value of the defined architecture are appropriately realized, it is necessary for continuing Architecture Governance of the implementation process to support design review, architecture refinement, and issue escalation.

Different classes of architecture engagement at different levels of the enterprise will require focus in specific areas, as shown below.

Engagement Type	Focus Iteration Cycles	Scope Focus
Supporting Business Strategy	Architecture Capability Architecture Development (Baseline First)	Broad, shallow consideration given to the Architecture Landscape in order to address a specific strategic question and define terms for more detailed architecture efforts to address strategy realization.
Architectural Portfolio Management of the Landscape	Architecture Capability Architecture Development (Baseline First)	Focus on physical assessment of baseline applications and technology infrastructure to identify improvement opportunities, typically within the constraints of maintaining business as usual.
Architectural Portfolio Management of Projects	Transition Planning Architecture Governance	Focus on projects, project dependencies, and landscape impacts to align project sequencing in a way that is architecturally optimized.
Architectural Definition of Foundational Change Initiatives	Architecture Capability Architecture Development (Baseline First) Transition Planning	Focus on elaborating a vision through definition of baseline and identifying what needs to change to transition the baseline to the target.

Engagement Type	Focus Iteration Cycles	Scope Focus
Architectural Definition of Bounded Change Initiatives	Architecture Development (Target First) Transition Planning	Focus on elaborating the target to meet a previously defined and agreed vision, scope, or set of constraints. Use the target as a basis for analysis to avoid perpetuation of baseline, sub-optimal architectures.
Architectural Governance of Change Implementation	Architecture Governance	Use the Architecture Vision, constraints, principles, requirements, Target Architecture definition, and transition roadmap to ensure that projects realize their intended benefit, are aligned with each other, and are aligned with wider business need.

18.4 Approaches to Architecture Development

Two approaches can be adopted within the ADM for the development of architectures:

- **Baseline First**: in this style, an assessment of the baseline landscape is used to identify problem areas and improvement opportunities

  This process is most suitable when the baseline is complex, not clearly understood, or agreed upon. This approach is common where organizational units have had a high degree of autonomy.

- **Target First**: in this style, the target solution is elaborated in detail and then mapped back to the baseline, in order to identify change activity

  This process is suitable when a target state is agreed at a high level and where the enterprise wishes to effectively transition to the target model.

Typically, if the baseline is broadly understood a higher value will be obtained focusing on the target first then baseline to the extent necessary to identify changes.

In practical terms, an architecture team will always give informal consideration to the baseline when analyzing the target (and *vice versa*). In situations where baseline and target are expected to be considered in parallel by stakeholders, it is recommended that the architecture team focuses priority on one state in order to maintain focus and consistency of execution.

18.5 Iteration Considerations

Some iteration cycles can be executed once, whereas others have a natural minimum number of cycles. For some iteration cycles, each iteration follows the same process; where there is more than one iteration within a cycle, the process differs slightly for each of the iterations.

When considering the usage of iteration cycles, it is also necessary to consider where to place appropriate checkpoints within the process. If the expected level of stakeholder involvement is high, it may be sensible to carry out very frequent but informal checkpoints to ensure that the process is moving in the intended direction. If stakeholders are less closely involved, then checkpoints may be less frequent but more formal. Checkpoints at the completion of each iteration cycle, or at the end of several iteration cycles, are common.

18.5.1 Iteration between ADM Cycles

Each iteration completes an ADM cycle at a single level of Architecture Description. This approach to the ADM uses Phase F (Migration Planning) to initiate new more detailed architecture development projects. This approach is illustrated in Figure 18-3. This type of iteration highlights the need for higher-level architecture to guide and constrain more detailed architecture. It also highlights that the complete Architecture Landscape is developed by multiple ADM iterations.

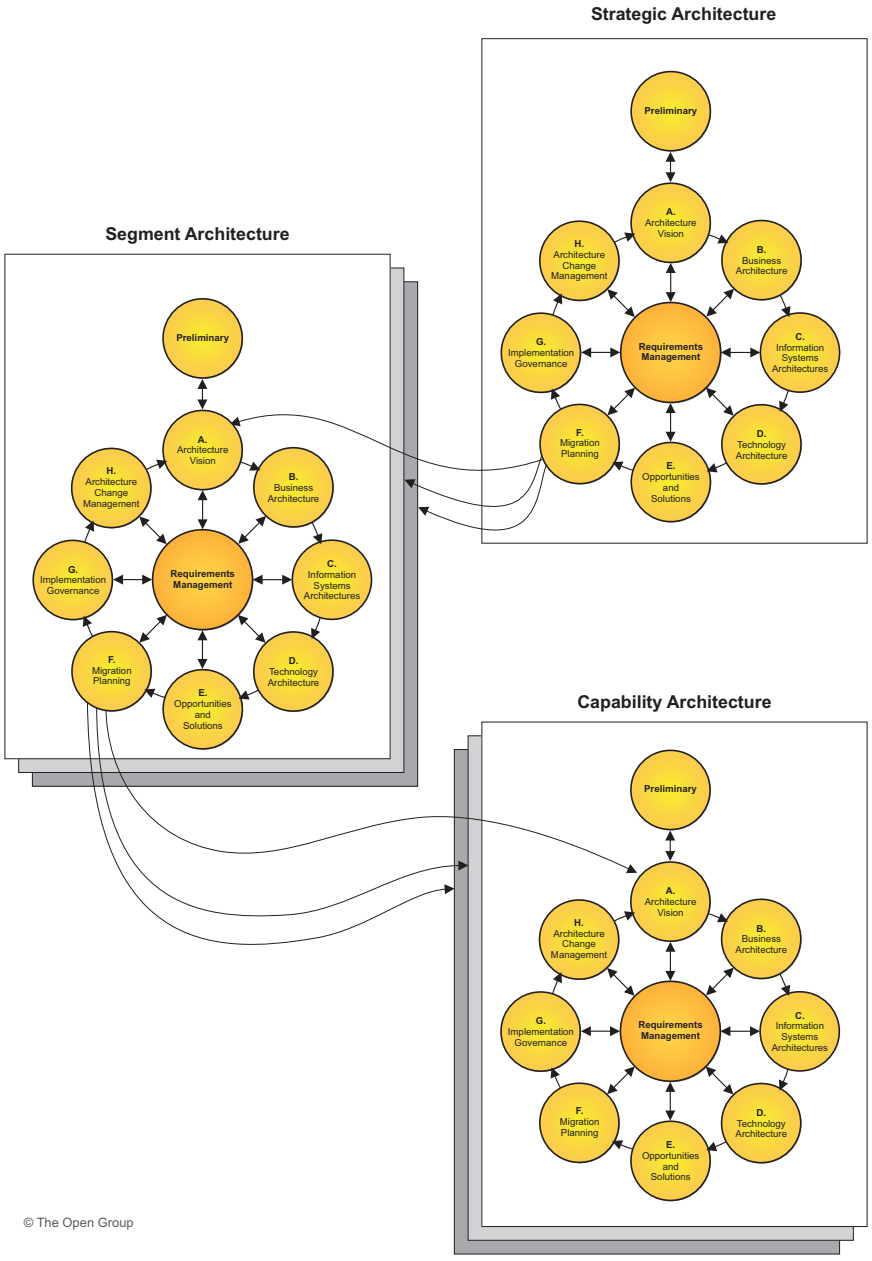

Figure 18-3 A Hierarchy of ADM Processes Example

18.5.2 Iteration within an ADM Cycle

Each iteration cycle crosses multiple TOGAF ADM phases. The following tables show at a high level which phases should be completed for which iteration cycle, showing activity that is core (i.e., the primary focus of the iteration), activity that is light (i.e., the secondary focus of the iteration), and activity that may be informally conducted (i.e., some activity may be carried out, but it is not explicitly mentioned in the ADM).

		Architecture Development			Transition Planning		Architecture Governance	
TOGAF Phase		Iteration 1	Iteration 2	Iteration *n*	Iteration 1	Iteration *n*	Iteration 1	Iteration *n*
Preliminary		Informal	Informal	Informal				Light
Architecture Vision		Informal	Informal	Informal	Informal	Informal		Light
Business Architecture	Baseline	Core	Light	Core	Informal	Informal		Light
	Target	Informal	Core	Core	Informal	Informal		Light
Application Architecture	Baseline	Core	Light	Core	Informal	Informal		Light
	Target	Informal	Core	Core	Informal	Informal		Light
Data Architecture	Baseline	Core	Light	Core	Informal	Informal		Light
	Target	Informal	Core	Core	Informal	Informal		Light
Technology Architecture	Baseline	Core	Light	Core	Informal	Informal		Light
	Target	Informal	Core	Core	Informal	Informal		Light
Opportunities and Solutions		Light	Light	Light	Core	Core	Informal	Informal
Migration Planning		Light	Light	Light	Core	Core	Informal	Informal
Implementation Governance					Informal	Informal	Core	Core
Change Management		Informal	Informal	Informal	Informal	Informal	Core	Core

Core: primary focus activity for the iteration

Light: secondary focus activity for the iteration

Informal: potential activity for the iteration, not formally mentioned in the method

Figure 18-4 Activity by Iteration for Baseline First Architecture Definition

		Architecture Development			Transition Planning		Architecture Governance	
TOGAF Phase		Iteration 1	Iteration 2	Iteration *n*	Iteration 1	Iteration *n*	Iteration 1	Iteration *n*
Preliminary		Informal	Informal	Informal				Light
Architecture Vision		Informal	Informal	Informal	Informal	Informal		Light
Business Architecture	Baseline	Informal	Core	Core	Informal	Informal		Light
	Target	Core	Light	Core	Informal	Informal		Light
Application Architecture	Baseline	Informal	Core	Core	Informal	Informal		Light
	Target	Core	Light	Core	Informal	Informal		Light
Data Architecture	Baseline	Informal	Core	Core	Informal	Informal		Light
	Target	Core	Light	Core	Informal	Informal		Light
Technology Architecture	Baseline	Informal	Core	Core	Informal	Informal		Light
	Target	Core	Light	Core	Informal	Informal		Light
Opportunities and Solutions		Light	Light	Light	Core	Core	Informal	Informal
Migration Planning		Light	Light	Light	Core	Core	Informal	Informal
Implementation Governance					Informal	Informal	Core	Core
Change Management		Informal	Informal	Informal	Informal	Informal	Core	Core

Core: primary focus activity for the iteration
Light: secondary focus activity for the iteration
Informal: potential activity for the iteration, not formally mentioned in the method

Figure 18-5 Activity by Iteration for Target First Architecture Definition

The suggested iteration cycles mapped to the TOGAF phases are described in the following table:

Iteration Cycle	Iteration	Purpose	Description
Architecture Development (Baseline First)	Iteration 1	Define the Baseline Architecture.	This iteration comprises a pass through the Business Architecture, Information Systems Architecture, and Technology Architecture phases of the ADM, focusing on definition of the baseline. Opportunities, solutions, and migration plans are also considered to drive out the focus for change and test feasibility.

Iteration Cycle	Iteration	Purpose	Description
	Iteration 2	Define the Target Architecture and gaps.	This iteration comprises a pass through the Business Architecture, Information Systems Architecture, and Technology Architecture phases of the ADM, focusing on definition of the target and analyzing gaps against the baseline. Opportunities, solutions, and migration plans are also considered to test viability.
	Iteration *n*	Refine baseline, target, and gaps.	Subsequent Architecture Development iterations attempt to correct and refine the target to achieve an outcome that is beneficial, feasible, and viable.
Architecture Development (Target First)	Iteration 1	Define the Target Architecture.	This iteration comprises a pass through the Business Architecture, Information Systems Architecture, and Technology Architecture phases of the ADM, focusing on definition of the target. Opportunities, solutions, and migration plans are also considered to drive out the focus for change and test feasibility.
	Iteration 2	Define the Baseline Architecture and gaps.	This iteration comprises a pass through the Business Architecture, Information Systems Architecture, and Technology Architecture phases of the ADM, focusing on definition of the baseline and analyzing gaps against the target. Opportunities, solutions, and migration plans are also considered to test viability.
	Iteration *n*	Refine baseline, target, and gaps.	Subsequent Architecture Development iterations attempt to correct and refine the target to achieve an outcome that is beneficial, feasible, and viable.

Iteration Cycle	Iteration	Purpose	Description
Transition Planning	Iteration 1	Define and agree a set of improvement opportunities, aligned against a provisional Transition Architecture.	The initial iteration of Transition Planning seeks to gain buy-in to a portfolio of solution opportunities in the Opportunities & Solutions phase of ADM. This iteration also delivers a provisional Migration Plan.
	Iteration *n*	Agree the Transition Architecture, refining the identified improvement opportunities to fit.	Subsequent iterations of Transition Planning seek to refine the Migration Plan, feeding back issues into the Opportunities & Solutions phase for refinement.
Architecture Governance	Iteration 1	Mobilize Architecture Governance and change management processes.	The initial Architecture Governance iteration establishes a process for governance of change and also puts in place the appropriate people, processes, and technology to support managed access to and change of the defined architecture.
	Iteration *n*	Carry out Architecture Governance and change control.	Subsequent iterations of the Architecture Governance cycle focus on periodic reviews of change initiatives to resolve issues and ensure compliance. Results of a Change Request may trigger another phase to be revisited; for example, feeding back a new requirement to the Preliminary Phase to improve the Architecture Capability, or a new requirement for the architecture into the Architecture Development phases.

18.6 Conclusions

All of these techniques are valid applications of the ADM. Combined together, they represent how the ADM can be used in practice. The ADM should always be used in an iterative process. How this process is exercised is dependent upon organizational factors. Particular factors for consideration include:

- **The formality and nature of established process checkpoints within the organization**

  Does the organization mandate that certain groups of activities are carried out between checkpoints? Does the organization mandate that certain activities must be finalized before other activities can be carried out?

- **The level of stakeholder involvement expected within the process**

  Are stakeholders expecting to be closely involved within the development of a solution, or are they expecting to see a complete set of deliverables for review and approval?

- **The number of teams involved and the relationships between different teams**

  Is the entire architecture being developed by a specific team, or is there a hierarchy of teams with governance relationships between them?

- **The maturity of the solution area and the expected amount of rework and refinement required to arrive at an acceptable solution**

  Can the solution be achieved in a single pass, or does it require extensive proof-of-concept and prototyping work to evolve a suitable outcome?

- **Attitude to risk**

  Does the organizational culture react negatively to partially complete work products being circulated? Does the organizational culture require solutions to be proved in a trial environment before they can be implemented for mainstream application?

- **The class of engagement**

  What is the context for development of the Enterprise Architecture?

Chapter 19

Applying the ADM Across the Architecture Landscape

19.1 Overview

In a typical enterprise, many architectures will be described in the Architecture Landscape at any point in time. Some architectures will address very specific needs; others will be more general. Some will address detail; some will provide a big picture. To address this complexity, the TOGAF standard uses the concepts of levels and the Enterprise Continuum to provide a conceptual framework for organizing the Architecture Landscape. These concepts are tightly linked with organizing actual content in the Architecture Repository and any architecture partitions discussed in Part V.

19.2 Architecture Landscape

Levels provide a framework for dividing the Architecture Landscape into three levels of granularity:

1. **Strategic Architecture** provides an organizing framework for operational and change activity and allows for direction setting at an executive level.
2. **Segment Architecture** provides an organizing framework for operational and change activity and allows for direction setting and the development of effective architecture roadmaps at a program or portfolio level.
3. **Capability Architecture** provides an organizing framework for change activity and the development of effective architecture roadmaps realizing capability increments.

Figure 19-1 shows a summary of the classification model for Architecture Landscapes.

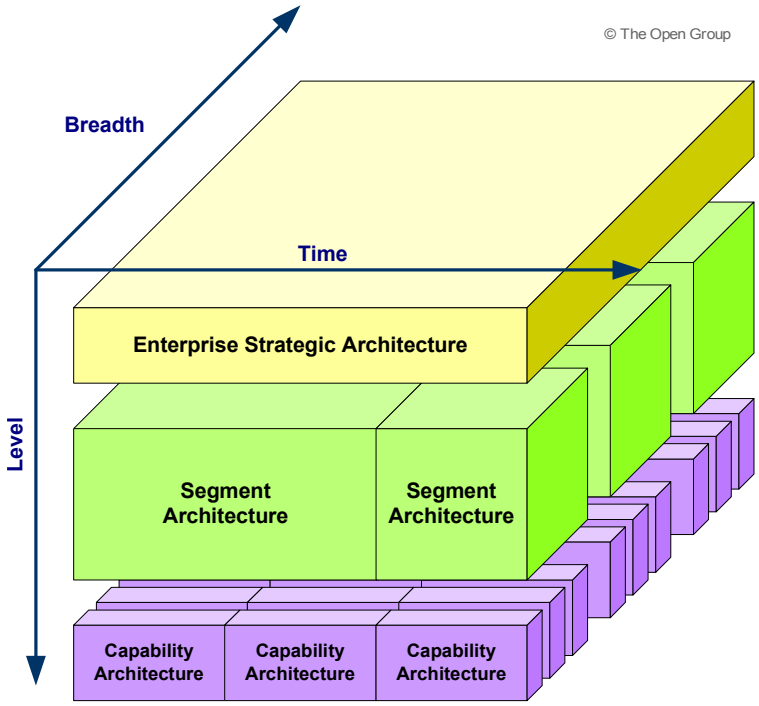

Figure 19-1 Summary Classification Model for Architecture Landscapes

The Architecture Continuum provides a method of dividing each level of the Architecture Landscape (see Section 35.4.1) by abstraction. It offers a consistent way to define and understand the generic rules, representations, and relationships in an architecture, including traceability and derivation relationships. The Architecture Continuum shows the relationships from foundation elements to organization-specific architecture, as shown in Figure 19-2.

The Architecture Continuum is a useful tool to discover commonality and eliminate unnecessary redundancy.

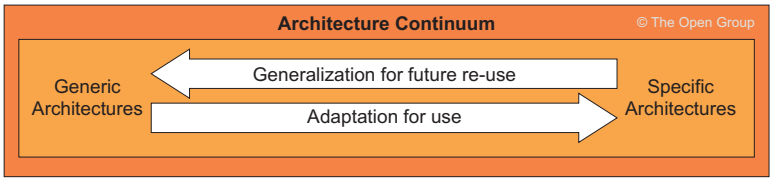

Figure 19-2 Summary of Architecture Continuum

Levels and the Architecture Continuum provide a comprehensive mechanism to describe and classify the Architecture Landscape. These concepts can be used to organize the Architecture Landscape into a set of related architectures with:

- Manageable complexity for each individual architecture or solution
- Defined groupings

- Defined hierarchies and navigation structures
- Appropriate processes, roles, and responsibilities attached to each grouping

There is no definitive organizing model for architecture, as each enterprise should adopt a model that reflects its own operating model.

19.3 Organizing the Architecture Landscape to Understand the State of the Enterprise

The following characteristics are typically used to organize the Architecture Landscape:

- **Breadth**: the breadth (subject matter) area is generally the primary organizing characteristic for describing an Architecture Landscape

  Architectures are functionally decomposed into a hierarchy of specific subject areas or segments.

- **Depth**: with broader subject areas, less detail is needed to ensure that the architecture has a manageable size and complexity

  More specific subject matter areas will generally permit (and require) more detailed architectures.

- **Time**: for a specific breadth and depth an enterprise can create a Baseline Architecture and a set of Target Architectures that stretch into the future

  Broader and less detailed architectures will generally be valid for longer periods of time and can provide a vision for the enterprise that stretches further into the future.

- **Recency**: finally, each architecture view will progress through a development cycle where it increases in accuracy until finally approved

  After approval, an architecture will begin to decrease in accuracy if not actively maintained. In some cases recency may be used as an organizing factor for historic architectures.

Using the criteria above, architectures can be grouped into Strategic, Segment, and Capability Architecture levels, as described in Figure 19-1.

19.4 Developing Architectures at Different Levels

The previous sections have identified that different types of architecture are required to address different stakeholder needs at different levels of the organization. Each architecture typically does not exist in isolation and must therefore sit within a governance hierarchy. Broad, summary architectures set the direction for narrow and detailed architectures.

A number of techniques can be employed to use the ADM as a process that supports such hierarchies of architectures. Essentially there are two strategies that can be applied:

1. Architectures at different levels can be developed through iterations within a single cycle of the ADM process
2. Architectures at different levels can be developed through a hierarchy of ADM processes, executed concurrently

At the extreme ends of the scale, either of these two options can be fully adopted. In practice, an architect is likely to need to blend elements of each to fit the exact requirements of their Request for Architecture Work. Each of these approaches is described in Chapter 18.

Chapter 20

Architecture Principles

This chapter describes principles for use in the development of an Enterprise Architecture.

20.1 Introduction

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.

Depending on the organization, principles may be established within different domains and at different levels. Two key domains inform the development and utilization of architecture:

- **Enterprise Principles** provide a basis for decision-making throughout an enterprise, and inform how the organization sets about fulfilling its mission

  Such principles are commonly found as a means of harmonizing decision-making across an organization. In particular, they are a key element in a successful Architecture Governance strategy (see Chapter 44).

  Within the broad domain of enterprise principles, it is common to have subsidiary principles within a business or organizational unit. Examples include IT, HR, domestic operations, or overseas operations. These principles provide a basis for decision-making within the subsidiary domain and will inform architecture development within the domain. Care must be taken to ensure that the principles used to inform architecture development align to the organizational context of the Architecture Capability.

- **Architecture Principles** are a set of principles that relate to architecture work

  They reflect a level of consensus across the enterprise, and embody the spirit and thinking of existing enterprise principles. Architecture Principles govern the architecture process, affecting the development, maintenance, and use of the Enterprise Architecture.

It is common to have sets of principles form a hierarchy, in that segment principles will be informed by, and elaborate on, the principles at the enterprise level. Architecture Principles will be informed and constrained by enterprise principles.

Architecture Principles may restate other enterprise guidance in terms and form that effectively guide architecture development.

The remainder of this section deals exclusively with Architecture Principles.

20.2 Characteristics of Architecture Principles

Architecture Principles define the underlying general rules and guidelines for the use and deployment of all IT resources and assets across the enterprise. They reflect a level of consensus among the various elements of the enterprise, and form the basis for making future IT decisions.

Each Architecture Principle should be clearly related back to the business objectives and key architecture drivers.

20.3 Components of Architecture Principles

It is useful to have a standard way of defining principles. In addition to a definition statement, each principle should have associated rationale and implications statements, both to promote understanding and acceptance of the principles themselves, and to support the use of the principles in explaining and justifying why specific decisions are made.

A recommended template is given in Table 20-1.

Name	Should both represent the essence of the rule as well as be easy to remember. Specific technology platforms should not be mentioned in the name or statement of a principle. Avoid ambiguous words in the Name and in the Statement such as: "support", "open", "consider", and for lack of good measure the word "avoid", itself, be careful with "manage(ment)", and look for unnecessary adjectives and adverbs (fluff).
Statement	Should succinctly and unambiguously communicate the fundamental rule. For the most part, the principles statements for managing information are similar from one organization to the next. It is vital that the principles statement is unambiguous.
Rationale	Should highlight the business benefits of adhering to the principle, using business terminology. Point to the similarity of information and technology principles to the principles governing business operations. Also describe the relationship to other principles, and the intentions regarding a balanced interpretation. Describe situations where one principle would be given precedence or carry more weight than another for making a decision.
Implications	Should highlight the requirements, both for the business and IT, for carrying out the principle — in terms of resources, costs, and activities/tasks. It will often be apparent that current systems, standards, or practices would be incongruent with the principle upon adoption. The impact to the business and consequences of adopting a principle should be clearly stated. The reader should readily discern the answer to: "How does this affect me?". It is important not to oversimplify, trivialize, or judge the merit of the impact. Some of the implications will be identified as potential impacts only, and may be speculative rather than fully analyzed.

Table 20-1 Recommended Format for Defining Principles

An example set of Architecture Principles following this template is given in Section 20.6.

20.4 Developing Architecture Principles

Architecture Principles are typically developed by the Enterprise Architects, in conjunction with the key stakeholders, and are approved by the Architecture Board.

Architecture Principles will be informed by principles at the enterprise level, if they exist.

Architecture Principles must be clearly traceable and clearly articulated to guide decision-making. They are chosen so as to ensure alignment of the architecture and implementation of the Target Architecture with business strategies and visions.

Specifically, the development of Architecture Principles is typically influenced by the following:

- **Enterprise mission and plans**: the mission, plans, and organizational infrastructure of the enterprise
- **Enterprise strategic initiatives**: the characteristics of the enterprise — its strengths, weaknesses, opportunities, and threats — and its current enterprise-wide initiatives (such as process improvement and quality management)
- **External constraints**: market factors (time-to-market imperatives, customer expectations, etc.); existing and potential legislation
- **Current systems and technology**: the set of information resources deployed within the enterprise, including systems documentation, equipment inventories, network configuration diagrams, policies, and procedures
- **Emerging industry trends**: predictions about economic, political, technical, and market factors that influence the enterprise environment

20.4.1 Qualities of Principles

Merely having a written statement that is called a principle does not mean that the principle is good, even if everyone agrees with it.

A good set of principles will be founded in the beliefs and values of the organization and expressed in language that the business understands and uses. Principles should be few in number, future-oriented, and endorsed and championed by senior management. They provide a firm foundation for making architecture and planning decisions, framing policies, procedures, and standards, and supporting resolution of contradictory situations. A poor set of principles will quickly become disused, and the resultant architectures, policies, and standards will appear arbitrary or self-serving, and thus lack credibility. Essentially, principles drive behavior.

There are five criteria that distinguish a good set of principles:

- **Understandable**: the underlying tenets can be quickly grasped and understood by individuals throughout the organization

  The intention of the principle is clear and unambiguous, so that violations, whether intentional or not, are minimized.

- **Robust**: enable good quality decisions about architectures and plans to be made, and enforceable policies and standards to be created

  Each principle should be sufficiently definitive and precise to support consistent decision-making in complex, potentially controversial situations.

- **Complete**: every potentially important principle governing the management of information and technology for the organization is defined — the principles cover every situation perceived

- **Consistent**: strict adherence to one principle may require a loose interpretation of another principle

  The set of principles must be expressed in a way that allows a balance of interpretations. Principles should not be contradictory to the point where adhering to one principle would violate the spirit of another. Every word in a principle statement should be carefully chosen to allow consistent yet flexible interpretation.

- **Stable**: principles should be enduring, yet able to accommodate changes

  An amendment process should be established for adding, removing, or altering principles after they are ratified initially.

20.5 Applying Architecture Principles

Architecture Principles are used to capture the fundamental truths about how the enterprise will use and deploy IT resources and assets. The principles are used in a number of different ways:

1. To provide a framework within which the enterprise can start to make conscious decisions about Enterprise Architecture and projects that implement the target Enterprise Architecture
2. As a guide to establishing relevant evaluation criteria, thus exerting strong influence on the selection of products, solutions, or solution architectures in the later stages of managing compliance to the Enterprise Architecture
3. As drivers for defining the functional requirements of the architecture
4. As an input to assessing both existing implementations and the strategic portfolio, for compliance with the defined architectures; these assessments will provide valuable insights into the transition activities needed to implement an architecture, in support of business goals and priorities
5. The Rationale statements within an Architecture Principle highlight the business value of implementations consistent with the principle and provide guidance for difficult decisions with conflicting drivers or objectives
6. The Implications statements within an Architecture Principle provide an outline of the key tasks, resources, and potential costs to the enterprise of following the principle; they also provide valuable inputs to future transition initiative and planning activities
7. Support the Architecture Governance activities in terms of:
   - — Providing a "back-stop" for the standard Architecture Compliance assessments where some interpretation is allowed or required
   - — Supporting the decision to initiate a dispensation request where the implications of a particular architecture amendment cannot be resolved within local operating procedure

Principles are inter-related, and need to be applied as a set.

Principles will sometimes compete; for example, the principles of "accessibility" and "security" tend towards conflicting decisions. Each principle must be considered in the context of "all other things being equal".

At times a decision will be required as to which principle will take precedence on a particular issue. The rationale for such decisions should always be documented.

A common reaction on first reading of a principle is "this is obvious and does not need to be

documented". The fact that a principle seems self-evident does not mean that the guidance in a principle is followed. Having principles that appear obvious helps ensure that decisions actually follow the desired outcome.

Although specific penalties are not prescribed in a declaration of principles, violations of principles generally cause operational problems and inhibit the ability of the organization to fulfil its mission.

20.6 Example Set of Architecture Principles

Too many principles can reduce the flexibility of the architecture. Many organizations prefer to define only high-level principles, and to limit the number to between 10 and 20.

The following example illustrates both the typical content of a set of Architecture Principles, and the recommended format for defining them, as explained above.

20.6.1 Business Principles

Principle 1: Primacy of Principles

Statement: These principles of information management apply to all organizations within the enterprise.

Rationale: The only way we can provide a consistent and measurable level of quality information to decision-makers is if all organizations abide by the principles.

Implications:

- Without this principle, exclusions, favoritism, and inconsistency would rapidly undermine the management of information
- Information management initiatives will not begin until they are examined for compliance with the principles
- A conflict with a principle will be resolved by changing the framework of the initiative

Principle 2: Maximize Benefit to the Enterprise

Statement: Information management decisions are made to provide maximum benefit to the enterprise as a whole.

Rationale: This principle embodies "service above self". Decisions made from an enterprise-wide perspective have greater long-term value than decisions made from any particular organizational perspective. Maximum return on investment requires information management decisions to adhere to enterprise-wide drivers and priorities. No minority group will detract from the benefit of the whole. However, this principle will not preclude any minority group from getting its job done.

Implications:

- Achieving maximum enterprise-wide benefit will require changes in the way we plan and manage information — technology alone will not bring about this change
- Some organizations may have to concede their own preferences for the greater benefit of the entire enterprise

- Application development priorities must be established by the entire enterprise for the entire enterprise
- Applications components should be shared across organizational boundaries
- Information management initiatives should be conducted in accordance with the enterprise plan

  Individual organizations should pursue information management initiatives which conform to the blueprints and priorities established by the enterprise. The plan will be changed as needed.
- As needs arise, priorities must be adjusted; a forum with comprehensive enterprise representation should make these decisions

Principle 3: Information Management is Everybody's Business

Statement: All organizations in the enterprise participate in information management decisions needed to accomplish business objectives.

Rationale: Information users are the key stakeholders, or customers, in the application of technology to address a business need. In order to ensure information management is aligned with the business, all organizations in the enterprise must be involved in all aspects of the information environment. The business experts from across the enterprise and the technical staff responsible for developing and sustaining the information environment need to come together as a team to jointly define the goals and objectives of IT.

Implications:

- To operate as a team, every stakeholder, or customer, will need to accept responsibility for developing the information environment
- Commitment of resources will be required to implement this principle

Principle 4: Business Continuity

Statement: Enterprise operations are maintained in spite of system interruptions.

Rationale: As system operations become more pervasive, we become more dependent on them; therefore, we must consider the reliability of such systems throughout their design and use. Business premises throughout the enterprise must be provided with the capability to continue their business functions regardless of external events. Hardware failure, natural disasters, and data corruption should not be allowed to disrupt or stop enterprise activities. The enterprise business functions must be capable of operating on alternative information delivery mechanisms.

Implications:

- Dependency on shared system applications mandates that the risks of business interruption must be established in advance and managed

  Management includes but is not limited to periodic reviews, testing for vulnerability and exposure, or designing mission-critical services to ensure business function continuity through redundant or alternative capabilities.
- Recoverability, redundancy, and maintainability should be addressed at the time of design

- Applications must be assessed for criticality and impact on the enterprise mission, in order to determine what level of continuity is required and what corresponding recovery plan is necessary

Principle 5: Common Use Applications

Statement: Development of applications used across the enterprise is preferred over the development of similar or duplicative applications which are only provided to a particular organization.

Rationale: Duplicative capability is expensive and proliferates conflicting data.

Implications:

- Organizations which depend on a capability which does not serve the entire enterprise must change over to the replacement enterprise-wide capability; this will require establishment of and adherence to a policy requiring this
- Organizations will not be allowed to develop capabilities for their own use which are similar/duplicative of enterprise-wide capabilities; in this way, expenditures of scarce resources to develop essentially the same capability in marginally different ways will be reduced
- Data and information used to support enterprise decision-making will be standardized to a much greater extent than previously

  This is because the smaller, organizational capabilities which produced different data (which was not shared among other organizations) will be replaced by enterprise-wide capabilities. The impetus for adding to the set of enterprise-wide capabilities may well come from an organization making a convincing case for the value of the data/information previously produced by its organizational capability, but the resulting capability will become part of the enterprise-wide system, and the data it produces will be shared across the enterprise.

Principle 6: Service Orientation

Statement: The architecture is based on a design of services which mirror real-world business activities comprising the enterprise (or inter-enterprise) business processes.

Rationale: Service orientation delivers enterprise agility and Boundaryless Information Flow.

Implications:

- Service representation utilizes business descriptions to provide context (i.e., business process, goal, rule, policy, service interface, and service component) and implements services using service orchestration
- Service orientation places unique requirements on the infrastructure, and implementations should use open standards to realize interoperability and location transparency
- Implementations are environment-specific; they are constrained or enabled by context and must be described within that context
- Strong governance of service representation and implementation is required

- A "Litmus Test", which determines a "good service", is required

Principle 7: Compliance with Law

Statement: Enterprise information management processes comply with all relevant laws, policies, and regulations.

Rationale: Enterprise policy is to abide by laws, policies, and regulations. This will not preclude business process improvements that lead to changes in policies and regulations.

Implications:

- The enterprise must be mindful to comply with laws, regulations, and external policies regarding the collection, retention, and management of data
- Education and access to the rules

  Efficiency, need, and common sense are not the only drivers. Changes in the law and changes in regulations may drive changes in our processes or applications.

Principle 8: IT Responsibility

Statement: The IT organization is responsible for owning and implementing IT processes and infrastructure that enable solutions to meet user-defined requirements for functionality, service levels, cost, and delivery timing.

Rationale: Effectively align expectations with capabilities and costs so that all projects are cost-effective. Efficient and effective solutions have reasonable costs and clear benefits.

Implications:

- A process must be created to prioritize projects
- The IT function must define processes to manage business unit expectations
- Data, application, and technology models must be created to enable integrated quality solutions and to maximize results

Principle 9: Protection of Intellectual Property

Statement: The enterprise's Intellectual Property (IP) must be protected. This protection must be reflected in the IT architecture, implementation, and governance processes.

Rationale: A major part of an enterprise's IP is hosted in the IT domain.

Implications:

- While protection of IP assets is everybody's business, much of the actual protection is implemented in the IT domain — even trust in non-IT processes can be managed by IT processes (email, mandatory notes, etc.)
- A security policy, governing human and IT actors, will be required that can substantially improve protection of IP; this must be capable of both avoiding compromises and reducing liabilities
- Resources on such policies can be found at the SANS Institute (refer to www.sans.org/security-resources/policies)

20.6.2 Data Principles

Principle 10: Data is an Asset

Statement: Data is an asset that has value to the enterprise and is managed accordingly.

Rationale: Data is a valuable corporate resource; it has real, measurable value. In simple terms, the purpose of data is to aid decision-making. Accurate, timely data is critical to accurate, timely decisions. Most corporate assets are carefully managed, and data is no exception. Data is the foundation of our decision-making, so we must also carefully manage data to ensure that we know where it is, can rely upon its accuracy, and can obtain it when and where we need it.

Implications:

- This is one of three closely-related principles regarding data: data is an asset; data is shared; and data is easily accessible

  The implication is that there is an education task to ensure that all organizations within the enterprise understand the relationship between value of data, sharing of data, and accessibility to data.

- Stewards must have the authority and means to manage the data for which they are accountable
- We must make the cultural transition from "data ownership" thinking to "data stewardship" thinking
- The role of data steward is critical because obsolete, incorrect, or inconsistent data could be passed to enterprise personnel and adversely affect decisions across the enterprise
- Part of the role of data steward, who manages the data, is to ensure data quality

  Procedures must be developed and used to prevent and correct errors in the information and to improve those processes that produce flawed information. Data quality will need to be measured and steps taken to improve data quality — it is probable that policy and procedures will need to be developed for this as well.

- A forum with comprehensive enterprise-wide representation should decide on process changes suggested by the steward
- Since data is an asset of value to the entire enterprise, data stewards accountable for properly managing the data must be assigned at the enterprise level

Principle 11: Data is Shared

Statement: Users have access to the data necessary to perform their duties; therefore, data is shared across enterprise functions and organizations.

Rationale: Timely access to accurate data is essential to improving the quality and efficiency of enterprise decision-making. It is less costly to maintain timely, accurate data in a single application, and then share it, than it is to maintain duplicative data in multiple applications. The enterprise holds a wealth of data, but it is stored in hundreds of incompatible stovepipe databases. The speed of data collection, creation, transfer, and assimilation is driven by the ability of the organization to efficiently share these islands of data across the organization.

Shared data will result in improved decisions since we will rely on fewer (ultimately one virtual) sources of more accurate and timely managed data for all of our decision-making. Electronically shared data will result in increased efficiency when existing data entities can be used, without re-keying, to create new entities.

Implications:

- This is one of three closely-related principles regarding data: data is an asset; data is shared; and data is easily accessible

  The implication is that there is an education task to ensure that all organizations within the enterprise understand the relationship between value of data, sharing of data, and accessibility to data.

- To enable data sharing we must develop and abide by a common set of policies, procedures, and standards governing data management and access for both the short and the long term
- For the short term, to preserve our significant investment in legacy systems, we must invest in software capable of migrating legacy system data into a shared data environment
- We will also need to develop standard data models, data elements, and other metadata that defines this shared environment and develop a repository system for storing this metadata to make it accessible
- For the long term, as legacy systems are replaced, we must adopt and enforce common data access policies and guidelines for new application developers to ensure that data in new applications remains available to the shared environment and that data in the shared environment can continue to be used by the new applications
- For both the short term and the long term we must adopt common methods and tools for creating, maintaining, and accessing the data shared across the enterprise
- Data sharing will require a significant cultural change
- This principle of data sharing will continually "bump up against" the principle of data security — under no circumstances will the data sharing principle cause confidential data to be compromised
- Data made available for sharing will have to be relied upon by all users to execute their respective tasks

  This will ensure that only the most accurate and timely data is relied upon for decision-making. Shared data will become the enterprise-wide "virtual single source" of data.

Principle 12: Data is Accessible

Statement: Data is accessible for users to perform their functions.

Rationale: Wide access to data leads to efficiency and effectiveness in decision-making, and affords a timely response to information requests and service delivery. Using information must be considered from an enterprise perspective to allow access by a wide variety of users. Staff time is saved and consistency of data is improved.

Implications:

- This is one of three closely-related principles regarding data: data is an asset; data is shared; and data is easily accessible

  The implication is that there is an education task to ensure that all organizations within the enterprise understand the relationship between value of data, sharing of data, and accessibility to data.

- Accessibility involves the ease with which users obtain information
- The way information is accessed and displayed must be sufficiently adaptable to meet a wide range of enterprise users and their corresponding methods of access
- Access to data does not constitute understanding of the data — personnel should take caution not to misinterpret information
- Access to data does not necessarily grant the user access rights to modify or disclose the data

  This will require an education process and a change in the organizational culture, which currently supports a belief in "ownership" of data by functional units.

Principle 13: Data Trustee

Statement: Each data element has a trustee accountable for data quality.

Rationale: One of the benefits of an architected environment is the ability to share data (e.g., text, video, sound, etc.) across the enterprise. As the degree of data sharing grows and business units rely upon common information, it becomes essential that only the data trustee makes decisions about the content of data. Since data can lose its integrity when it is entered multiple times, the data trustee will have sole responsibility for data entry which eliminates redundant human effort and data storage resources.

Note: A trustee is different than a steward — a trustee is responsible for accuracy and currency of the data, while responsibilities of a steward may be broader and include data standardization and definition tasks.

Implications:

- Real trusteeship dissolves the data "ownership" issues and allows the data to be available to meet all users' needs

  This implies that a cultural change from data "ownership" to data "trusteeship" may be required.

- The data trustee will be responsible for meeting quality requirements levied upon the data for which the trustee is accountable
- It is essential that the trustee has the ability to provide user confidence in the data based upon attributes such as "data source"
- It is essential to identify the true source of the data in order that the data authority can be assigned this trustee responsibility

  This does not mean that classified sources will be revealed nor does it mean the source will be the trustee.

- Information should be captured electronically once and immediately validated as close to the source as possible

  Quality control measures must be implemented to ensure the integrity of the data.

- As a result of sharing data across the enterprise, the trustee is accountable and responsible for the accuracy and currency of their designated data element(s) and, subsequently, must then recognize the importance of this trusteeship responsibility

Principle 14: Common Vocabulary and Data Definitions

Statement: Data is defined consistently throughout the enterprise, and the definitions are understandable and available to all users.

Rationale: The data that will be used in the development of applications must have a common definition throughout the Headquarters to enable sharing of data. A common vocabulary will facilitate communications and enable dialog to be effective. In addition, it is required to interface systems and exchange data.

Implications:

- We are lulled into thinking that this issue is adequately addressed because there are people with "data administration" job titles and forums with charters implying responsibility

  Significant additional energy and resources must be committed to this task. It is key to the success of efforts to improve the information environment. This is separate from but related to the issue of data element definition, which is addressed by a broad community — this is more like a common vocabulary and definition.

- The enterprise must establish the initial common vocabulary for the business; the definitions will be used uniformly throughout the enterprise

- Whenever a new data definition is required, the definition effort will be co-ordinated and reconciled with the corporate "glossary" of data descriptions

  The enterprise data administrator will provide this co-ordination.

- Ambiguities resulting from multiple parochial definitions of data must give way to accepted enterprise-wide definitions and understanding

- Multiple data standardization initiatives need to be co-ordinated

- Functional data administration responsibilities must be assigned

Principle 15: Data Security

Statement: Data is protected from unauthorized use and disclosure. In addition to the traditional aspects of national security classification, this includes, but is not limited to, protection of pre-decisional, sensitive, source selection-sensitive, and proprietary information.

Rationale: Open sharing of information and the release of information via relevant legislation must be balanced against the need to restrict the availability of classified, proprietary, and sensitive information.

Existing laws and regulations require the safeguarding of national security and the privacy of data, while permitting free and open access. Pre-decisional (work-in-progress, not yet authorized for release) information must be protected to avoid unwarranted speculation, misinterpretation, and inappropriate use.

Implications:

- Aggregation of data, both classified and not, will create a large target requiring review and de-classification procedures to maintain appropriate control

  Data owners and/or functional users must determine whether the aggregation results in an increased classification level. Appropriate policy and procedures will be needed to handle this review and de-classification. Access to information based on a need-to-know policy will force regular reviews of the body of information.

- The current practice of having separate systems to contain different classifications needs to be rethought

  Is there a software solution to separating classified and unclassified data? The current hardware solution is unwieldy, inefficient, and costly. It is more expensive to manage unclassified data on a classified system. Currently, the only way to combine the two is to place the unclassified data on the classified system, where it must remain.

- In order to adequately provide access to open information while maintaining secure information, security needs must be identified and developed at the data level, not the application level

- Data security safeguards can be put in place to restrict access to "view only" or "never see"

  Sensitivity labeling for access to pre-decisional, decisional, classified, sensitive, or proprietary information must be determined.

- Security must be designed into data elements from the beginning; it cannot be added later

  Systems, data, and technologies must be protected from unauthorized access and manipulation. Headquarters information must be safeguarded against inadvertent or unauthorized alteration, sabotage, disaster, or disclosure.

- New policies are needed on managing duration of protection for pre-decisional information and other works-in-progress, in consideration of content freshness

20.6.3 Application Principles

Principle 16: Technology Independence

Statement: Applications are independent of specific technology choices and therefore can operate on a variety of technology platforms.

Rationale: Independence of applications from the underlying technology allows applications to be developed, upgraded, and operated in the most cost-effective and timely way. Otherwise technology, which is subject to continual obsolescence and vendor dependence, becomes the driver rather than the user requirements themselves.

Realizing that every decision made with respect to IT makes us dependent on that technology, the intent of this principle is to ensure that Application Software is not dependent on specific hardware and operating systems software.

Implications:

- This principle will require standards which support portability
- For Commercial Off-The-Shelf (COTS) and Government Off-The-Shelf (GOTS) applications, there may be limited current choices, as many of these applications are technology and platform-dependent
- Subsystem interfaces will need to be developed to enable legacy applications to interoperate with applications and operating environments developed under the Enterprise Architecture
- Middleware should be used to decouple applications from specific software solutions
- As an example, this principle could lead to use of Java, and future Java-like protocols, which give a high degree of priority to platform-independence

Principle 17: Ease-of-Use

Statement: Applications are easy to use. The underlying technology is transparent to users, so they can concentrate on tasks at hand.

Rationale: The more a user has to understand the underlying technology, the less productive that user is. Ease-of-use is a positive incentive for use of applications. It encourages users to work within the integrated information environment instead of developing isolated systems to accomplish the task outside of the enterprise's integrated information environment. Most of the knowledge required to operate one system will be similar to others. Training is kept to a minimum, and the risk of using a system improperly is low.

Using an application should be as intuitive as driving a different car.

Implications:

- Applications will be required to have a common "look-and-feel" and support ergonomic requirements; hence, the common look-and-feel standard must be designed and usability test criteria must be developed
- Guidelines for user interfaces should not be constrained by narrow assumptions about user location, language, systems training, or physical capability

  Factors such as linguistics, customer physical infirmities (visual acuity, ability to use keyboard/mouse), and proficiency in the use of technology have broad ramifications in determining the ease-of-use of an application.

20.6.4 Technology Principles

Principle 18: Requirements-Based Change

Statement: Only in response to business needs are changes to applications and technology made.

Rationale: This principle will foster an atmosphere where the information environment changes in response to the needs of the business, rather than having the business change in response to IT changes. This is to ensure that the purpose of the information support — the transaction of business — is the basis for any proposed change.

Unintended effects on business due to IT changes will be minimized.

A change in technology may provide an opportunity to improve the business process and, hence, change business needs.

Implications:

- Changes in implementation will follow full examination of the proposed changes using the Enterprise Architecture
- There is no funding for a technical improvement or system development unless a documented business need exists
- Change management processes conforming to this principle will be developed and implemented
- This principle may bump up against the responsive change principle

  We must ensure the requirements documentation process does not hinder responsive change to meet legitimate business needs. The purpose of this principle is to keep the focus on business, not technology needs — responsive change is also a business need.

Principle 19: Responsive Change Management

Statement: Changes to the enterprise information environment are implemented in a timely manner.

Rationale: If people are to be expected to work within the enterprise information environment, that information environment must be responsive to their needs.

Implications:

- Processes for managing and implementing change must be developed that do not create delays
- A user who feels a need for change will need to connect with a "business expert" to facilitate explanation and implementation of that need
- If changes are going to be made, the architectures must be kept updated
- Adopting this principle might require additional resources
- This will conflict with other principles (e.g., maximum enterprise-wide benefit, enterprise-wide applications, etc.)

Principle 20: Control Technical Diversity

Statement: Technological diversity is controlled to minimize the non-trivial cost of maintaining expertise in and connectivity between multiple processing environments.

Rationale: There is a real, non-trivial cost of infrastructure required to support alternative technologies for processing environments. There are further infrastructure costs incurred to keep multiple processor constructs interconnected and maintained.

Limiting the number of supported components will simplify maintainability and reduce costs.

The business advantages of minimum technical diversity include: standard packaging of components; predictable implementation impact; predictable valuations and returns; redefined testing; utility status; and increased flexibility to accommodate technological advancements. Common technology across the enterprise brings the benefits of economies of scale to the enterprise. Technical administration and support costs are better controlled when limited resources can focus on this shared set of technology.

Implications:

- Policies, standards, and procedures that govern acquisition of technology must be tied directly to this principle
- Technology choices will be constrained by the choices available within the technology blueprint

  Procedures for augmenting the acceptable technology set to meet evolving requirements will have to be developed and put in place.
- The technology baseline is not being frozen

  Technology advances are welcomed and will change the technology blueprint when compatibility with the current infrastructure, improvement in operational efficiency, or a required capability has been demonstrated.

Principle 21: Interoperability

Statement: Software and hardware should conform to defined standards that promote interoperability for data, applications, and technology.

Rationale: Standards help ensure consistency, thus improving the ability to manage systems and improve user satisfaction, and protect existing IT investments, thus maximizing return on investment and reducing costs. Standards for interoperability additionally help ensure support from multiple vendors for their products, and facilitate supply chain integration.

Implications:

- Interoperability standards and industry standards will be followed unless there is a compelling business reason to implement a non-standard solution
- A process for setting standards, reviewing and revising them periodically, and granting exceptions must be established
- The existing IT platforms must be identified and documented

Chapter 21

Stakeholder Management

21.1 Introduction

Stakeholder management is an important discipline that successful architecture practitioners can use to win support from others. It helps them ensure that their projects succeed where others fail.

The benefits of successful stakeholder management are that:

- The most powerful stakeholders can be identified early and their input can then be used to shape the architecture; this ensures their support and improves the quality of the models produced
- Support from the more powerful stakeholders will help the engagement win more resources, thus making the architecture engagement more likely to succeed
- By communicating with stakeholders early and frequently, the architecture team can ensure that they fully understand the architecture process, and the benefits of Enterprise Architecture; this means they can support the architecture team more actively when necessary
- The architecture team can more effectively anticipate likely reactions to the architecture models and reports, and can build into the plan the actions that will be needed to capitalize on positive reaction while avoiding or addressing any negative reactions
- The architecture team can identify conflicting or competing objectives among stakeholders early and develop a strategy to resolve the issues arising from them

It is essential in any initiative to identify the individuals and groups within the organization who will contribute to the development of the architecture, identify those that will gain and those that will lose from its introduction, and then develop a strategy for dealing with them.

21.2 Approach to Stakeholder Management

Stakeholder analysis should be used during Phase A (Architecture Vision) to identify the key players in the engagement, and also be updated throughout each phase; different stakeholders may be uncovered as the engagement progresses through into Opportunities & Solutions, Migration Planning, and Architecture Change Management.

Complex architectures are extremely hard to manage, not only in terms of the architecture development process itself, but also in terms of obtaining agreement from the large numbers of stakeholders touched by it.

For example, just as a building architect will create wiring diagrams, floor plans, and elevations to describe different facets of a building to its different stakeholders (electricians, owners,

planning officials), so an Enterprise Architect must create different architecture views of the Business, Information Systems, and Technology Architecture for the stakeholders who have concerns related to these aspects.

The TOGAF standard specifically identifies this issue throughout the ADM through the following concepts (see Section 31.1):

- Architecture View
- Architecture Viewpoint
- Concern
- Stakeholder

21.3 Steps in the Stakeholder Management Process

The following sections detail recommended stakeholder management activity.

21.3.1 Identify Stakeholders

Identify the key stakeholders of the Enterprise Architecture.

The first task is to brainstorm who the main Enterprise Architecture stakeholders are. As part of this, think of all the people who are affected by it, who have influence or power over it, or have an interest in its successful or unsuccessful conclusion.

It might include senior executives, project organization roles, client organization roles, system developers, alliance partners, suppliers, IT operations, customers, etc.

When identifying stakeholders there is a danger of concentrating too heavily on the formal structure of an organization as the basis for identification. Informal stakeholder groups may be just as powerful and influential as the formal ones.

Most individuals will belong to more than one stakeholder group, and these groups tend to arise as a result of specific events.

Look at who is impacted by the Enterprise Architecture project:

- Who gains and who loses from this change?
- Who controls change management of processes?
- Who designs new systems?
- Who will make the decisions?
- Who procures IT systems and who decides what to buy?
- Who controls resources?
- Who has specialist skills the project needs?
- Who has influence?

In particular, influencers need to be identified. These will be well respected and moving up, participate in important meetings and committees (look at meeting minutes), know what's going on in the company, be valued by their peers and superiors, and not necessarily be in any formal position of power.

Although stakeholders may be both organizations and people, ultimately the Enterprise

Architecture team will need to communicate with people. It is the correct individual stakeholders within a stakeholder organization that need to be formally identified.

21.3.1.1 Sample Stakeholder Analysis

A sample stakeholder analysis that distinguishes 22 types of stakeholder, in five broad categories, is shown in Figure 21-1. Any particular architecture project may have more, fewer, or different stakeholders; and they may be grouped into more, fewer, or different categories.

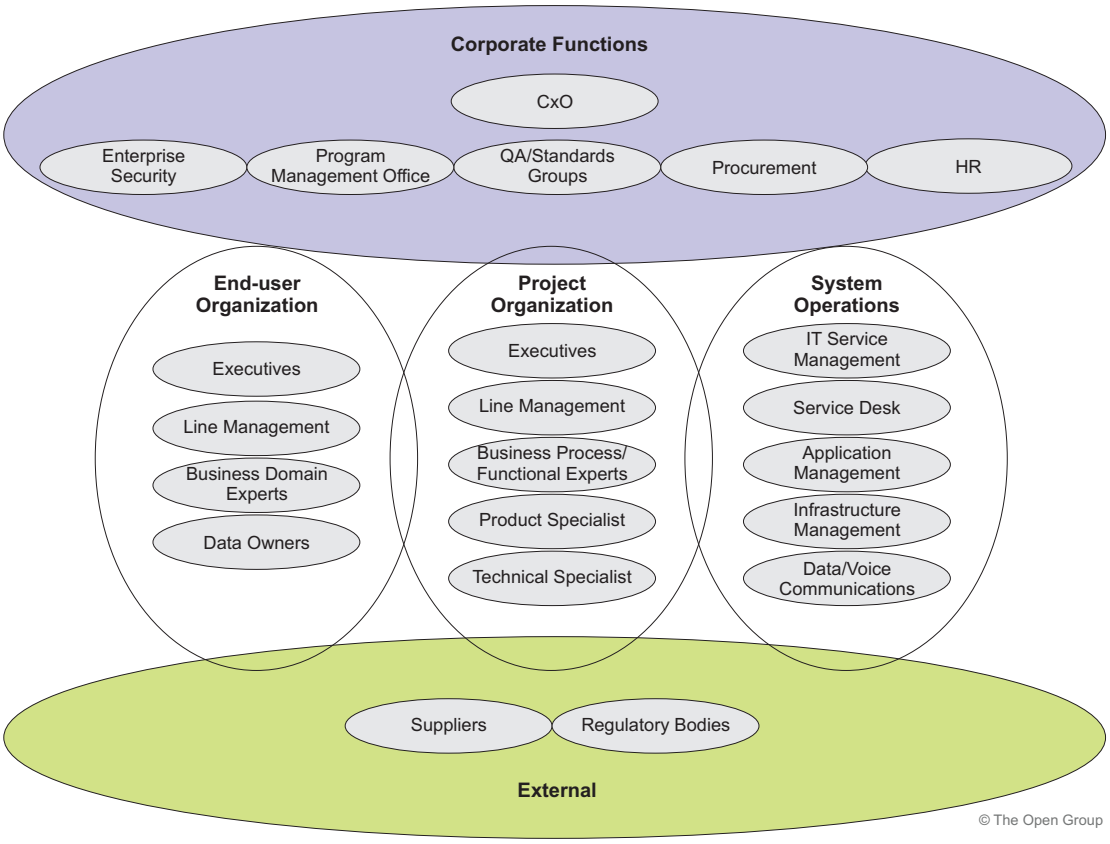

Figure 21-1 Sample Stakeholders and Categories

Consider both the Visible team — those obviously associated with the project/change — and the Invisible team — those who must make a real contribution to the project/change for it to be successful but who are not obviously associated with it (e.g., providers of support services).

21.3.2 Classify Stakeholder Positions

Develop a good understanding of the most important stakeholders and record this analysis for reference and refresh during the project. An example stakeholder analysis is shown in Table 21-1.

Stakeholder Group	Stakeholder	Ability to Disrupt Change	Current Under-standing	Required Under-standing	Current Commit-ment	Required Commit-ment	Required Support
CIO	John Smith	H	M	H	L	M	H
CFO	Jeff Brown	M	M	M	L	M	M

Table 21-1 Example Stakeholder Analysis

It is also important to assess the readiness of each stakeholder to behave in a supportive manner (i.e., demonstrate commitment to the Enterprise Architecture initiative).

This can be done by asking a series of questions:

- Is that person ready to change direction and begin moving towards the Target Architecture? If so, how ready?
- Is that person capable of being a credible advocate or agent of the proposed Enterprise Architecture initiative? If so, how capable?
- How involved is the individual in the Enterprise Architecture initiative? Are they simply an interested observer, or do they need to be involved in the details?
- Has that person made a contractual commitment to the development of the Enterprise Architecture, and its role in the governance of the development of the organization?

Then, for each person whose commitment is critical to ensure success, make a judgment as to their current level of commitment and the desired future level of commitment.

21.3.3 Determine Stakeholder Management Approach

The previous steps identified a long list of people and organizations that are affected by the Enterprise Architecture project.

Some of these may have the power either to block or advance. Some may be interested in what the Enterprise Architecture initiative is doing; others may not care. This step enables the team to easily see which stakeholders are expected to be blockers or critics, and which stakeholders are likely to be advocates and supporters of the initiative.

Work out stakeholder power, influence, and interest, so as to focus the Enterprise Architecture engagement on the key individuals. These can be mapped onto a power/interest matrix, which also indicates the strategy to adopt for engaging with them. Figure 21-2 shows an example power grid matrix.

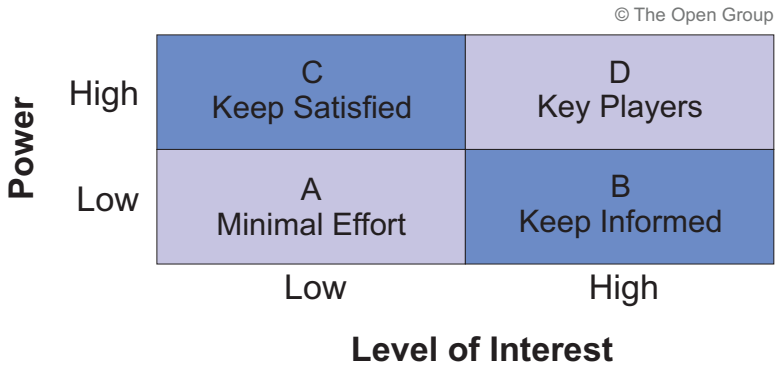

Figure 21-2 Stakeholder Power Grid

21.3.4 Tailor Engagement Deliverables

Identify catalogs, matrices, and diagrams that the architecture engagement needs to produce and validate with each stakeholder group to deliver an effective architecture model.

It is important to pay particular attention to stakeholder interests by defining specific catalogs, matrices, and diagrams that are relevant for a particular Enterprise Architecture model. This enables the architecture to be communicated to, and understood by, all the stakeholders, and enables them to verify that the Enterprise Architecture initiative will address their concerns.

21.4 Template Stakeholder Map

The following table provides an example stakeholder map for a TOGAF architecture project which has stakeholders as identified in Figure 21-1.

Stakeholder	Key Concerns	Class	Catalogs, Matrices, and Diagrams
CxO (Corporate Functions); e.g., CEO, CFO, CIO, COO	The high-level drivers, goals, and objectives of the organization, and how these are translated into an effective process and IT architecture to advance the business.	KEEP SATISFIED	Business Footprint diagram Goal/Objective/ Service diagram Organization Decomposition diagram Business Capabilities catalog Capability/Organization matrix Business Capability Map Strategy/Capability matrix Capability/Organization matrix Business Model diagram Value Stream catalog Value Stream Stages catalog Value Stream/Capability matrix Value Stream Map

Stakeholder	Key Concerns	Class	Catalogs, Matrices, and Diagrams
Program Management Office (Corporate Functions); e.g., Project Portfolio Managers	Prioritizing, funding, and aligning change activity. An understanding of project content and technical dependencies between projects supports portfolio management decision-making.	KEEP SATISFIED	Requirements catalog Project Context diagram Benefits diagram Business Footprint diagram Application Communication diagram Functional Decomposition diagram Business Capabilities catalog Capability/Organization matrix Business Capability Map Strategy/Capability matrix Capability/Organization matrix Business Model diagram Value Stream catalog Value Stream Stages catalog Value Stream/Capability matrix Value Stream Map
Procurement (Corporate Functions); e.g., Acquirers	Understanding what building blocks of the architecture can be bought, and what constraints (or rules) are relevant to the purchase. Acquirers will shop with multiple vendors looking for the best cost solution while adhering to the constraints (or rules) derived from the architecture, such as standards. The key concern is to make purchasing decisions that fit the architecture.	KEY PLAYERS	Technology Portfolio catalog Technology Standards catalog

Stakeholder	Key Concerns	Class	Catalogs, Matrices, and Diagrams
Human Resources (HR) (Corporate Functions); e.g., HR Managers, Training & Development Managers	The roles and actors are required to support the architecture and changes to it. The key concern is managing people transitions.	KEEP INFORMED	Organization Decomposition diagram Organization/Actor catalog Location catalog Application and User Location diagram Business Capabilities catalog Capability/Organization matrix Business Capability Map Strategy/Capability matrix Capability/Organization matrix Business Model diagram
Enterprise Security (Corporate Functions); e.g., Corporate Risk Management, Security Officers, IT Security Managers	Ensuring that the information, data, and systems of the organization are available to only those that have permission, and protecting the information, data, and systems from unauthorized tampering.	KEY PLAYERS	Product Lifecycle diagram Data Dissemination diagram Data Security diagram Actor/Role matrix Networked Computing Hardware diagram Network and Communications diagram

Stakeholder	Key Concerns	Class	Catalogs, Matrices, and Diagrams
QA/Standards Group (Corporate Functions); e.g., Data Owners, Process Owners, Technical Standards Bodies	Ensuring the consistent governance of the organization's business, data, application, and technology assets.	KEY PLAYERS	Process/Event/ Control/Product catalog Contract/Measure catalog Application Portfolio catalog Interface catalog Technology Standards catalog Technology Portfolio catalog Value Stream catalog Value Stream Stages catalog Value Stream/Capability matrix Value Stream Map
Executive (End-user Organization); e.g., Business Unit Directors, Business Unit CxOs, Business Unit Head of IT/Architecture	The high-level drivers, goals, and objectives of the organization, and how these are translated into an effective process and architecture to advance the business.	KEEP SATISFIED	Business Footprint diagram Goal/Objective/ Service diagram Organization Decomposition diagram Process Flow diagram Application Communication diagram Business Capabilities catalog Capability/Organization matrix Business Capability Map Strategy/Capability matrix Capability/Organization matrix Business Model diagram Value Stream catalog Value Stream Stages catalog Value Stream/Capability matrix Value Stream Map

Stakeholder	Key Concerns	Class	Catalogs, Matrices, and Diagrams
Line Management (End-user Organization); e.g., Senior Business Managers, Operations Regional Managers, IT Managers	Top-level functions and processes of the organization, and how the key applications support these processes.	KEY PLAYERS	Business Footprint diagram Organization Decomposition diagram Functional Decomposition diagram Process Flow diagram Application Communication diagram Application and User Location diagram Business Capabilities catalog Capability/Organization matrix Business Capability Map Strategy/Capability matrix Capability/Organization matrix Business Model diagram Value Stream catalog Value Stream Stages catalog Value Stream/Capability matrix Value Stream Map

Stakeholder	Key Concerns	Class	Catalogs, Matrices, and Diagrams
Business Domain Experts (End-user Organization); e.g., Business Process Experts, Business/Process Analyst, Process Architect, Process Designer, Functional Managers, Business Analyst	Functional aspects of processes and supporting systems. This can cover the human actors involved in the system, the user processes involved in the system, the functions required to support the processes, and the information required to flow in support of the processes.	KEY PLAYERS	Business Interaction matrix Actor/Role matrix Business Service/ Information diagram Functional Decomposition diagram Product Lifecycle diagram Business Use-Case diagram Application Use-Case diagram Application Communication diagram Data Entity/Business Function matrix Value Stream catalog Value Stream Stages catalog Value Stream/Capability matrix Value Stream Map
IT Service Management (Systems Operations); e.g., Service Delivery Manager	Ensuring that IT services provided to the organization meet the service levels required by that organization to succeed in business.	KEEP INFORMED	Technology Standards catalog Technology Portfolio catalog Contract/Measure catalog Process/Application Realization diagram Enterprise Manageability diagram

Stakeholder	Key Concerns	Class	Catalogs, Matrices, and Diagrams
IT Operations — Applications (System Operations); e.g., Application Architecture, System & Software Engineers	Development approach, software modularity and re-use, portability migration, and interoperability.	KEY PLAYERS	Process/Application Realization diagram Application/Data matrix Application Migration diagram Software Engineering diagram Platform decomposition Diagram Networked Computing/ Hardware diagram Software distribution Diagram
IT Operations — Infrastructure (System Operations); e.g., Infrastructure Architect, Wintel support, Mid-range support, Operational DBA, Service Desk	Location, modifiability, re-usability, and availability of all components of the system. Ensuring that the appropriate components are developed and deployed within the system in an optimal manner.	KEY PLAYERS	Platform Decomposition diagram Technology Standards catalog Technology Portfolio catalog Enterprise Manageability diagram Networked Computing/ Hardware diagram Processing diagram Environments and Locations diagram
IT Operations — Data/Voice Communications (System Operations); e.g., Network Management	Location, modifiability, re-usability, and availability of communications and networking services. Ensuring that the appropriate communications and networking services are developed and deployed within the system in an optimal manner.	KEY PLAYERS	Network and Communications diagram

Stakeholder	Key Concerns	Class	Catalogs, Matrices, and Diagrams
Executive (Project Organization); e.g., Sponsor, Program Manager	On-time, on-budget delivery of a change initiative that will realize expected benefits for the organization.	KEEP INFORMED	Requirements catalog Principles catalog Value Chain diagram Solution Concept diagram Functional Decomposition diagram Application and User Location diagram Business Capabilities catalog Capability/Organization matrix Business Capability Map Strategy/Capability matrix Capability/Organization matrix Business Model diagram Value Stream catalog Value Stream Stages catalog Value Stream/Capability matrix Value Stream Map

Stakeholder	Key Concerns	Class	Catalogs, Matrices, and Diagrams
Line Management (Project Organization); e.g., Project Manager	Operationally achieving on-time, on-budget delivery of a change initiative with an agreed scope.	KEEP INFORMED	Application Communication diagram Functional Decomposition diagram Environments and Locations diagram Business Capabilities catalog Capability/Organization matrix Business Capability Map Strategy/Capability matrix Capability/Organization matrix Business Model diagram Value Stream catalog Value Stream Stages catalog Value Stream/Capability matrix Value Stream Map

Stakeholder	Key Concerns	Class	Catalogs, Matrices, and Diagrams
Business Process/Functional Expert (Project Organization); e.g., Financials FICO® Functional Consultant, HR Functional Consultant	Adding more detail to the functional requirements of a change initiative based on experience and interaction with business domain experts in the end-user organization.	KEY PLAYERS	Process Flow diagram Business Use-Case diagram Business Service/Information diagram Functional Decomposition diagram Application Communication diagram Business Capabilities catalog Capability/Organization matrix Business Capability Map Strategy/Capability matrix Capability/Organization matrix Business Model diagram Value Stream catalog Value Stream Stages catalog Value Stream/Capability matrix Value Stream Map
Product Specialist (Project Organization); e.g., Portal Product Specialist	Specifying technology product designs in order to meet project requirements and comply with the Architecture Vision of the solution. In a packages and packaged services environment, product expertise can be used to identify product capabilities that can be readily leveraged and can provide guidance on strategies for product customization.	KEY PLAYERS	Software Engineering diagram Application/Data matrix

Stakeholder	Key Concerns	Class	Catalogs, Matrices, and Diagrams
Technical Specialist (Project Organization); e.g., Application Architect	Specifying technology product designs in order to meet project requirements and comply with the Architecture Vision of the solution.	KEY PLAYERS	Software Engineering diagram Platform Decomposition diagram Process/Application Realization diagram Application/Data matrix Application Migration diagram
Regulatory Bodies (Outside Services); e.g., Financial Regulator, Industry Regulator	Receipt of the information they need in order to regulate the client organization, and ensuring that their information requirements are properly satisfied. Interested in reporting processes, and the data and applications used to provide regulatory return information.	KEEP SATISFIED	Business Footprint diagram Application Communication diagram
Suppliers (Outside Services); e.g., Alliance Partners, Key Suppliers	Ensuring that their information exchange requirements are met in order that agreed service contracts with the client organizations can be fulfilled.	KEEP SATISFIED	Business Footprint diagram Business Service/Information diagram Application Communication diagram Business Capabilities catalog Capability/Organization matrix Business Capability Map Strategy/Capability matrix Capability/Organization matrix Business Model diagram Value Stream catalog Value Stream Stages catalog Value Stream/Capability matrix Value Stream Map

Chapter 22

Architecture Patterns

This chapter provides guidelines for using architecture patterns.

22.1 Introduction

Patterns for describing Enterprise Architectures are becoming increasingly important to practitioners. The diverse and multi-disciplinary nature of Enterprise Architecture requires that patterns be developed in different disciplines, domains, and levels of detail.

Previous versions of this standard did not fully embrace architecture patterns due to their perceived lack of maturity. Today, many organizations are using patterns to describe their architectures at various levels ranging from software design patterns to business patterns. It remains true that there is no single standard for describing Enterprise Architecture patterns. However, it can be said that there is a pattern for describing patterns.

22.1.1 Background

A "pattern" has been defined as: "an idea that has been useful in one practical context and will probably be useful in others" (Source: Analysis Patterns — Re-usable Object Models, by M. Fowler).

In the TOGAF standard, patterns are considered to be a way of putting building blocks into context; for example, to describe a re-usable solution to a problem. Building blocks are what you use: patterns can tell you how you use them, when, why, and what trade-offs you have to make in doing so.

Patterns offer the promise of helping the architect to identify combinations of Architecture and/or Solution Building Blocks (ABBs/SBBs) that have been proven to deliver effective solutions in the past, and may provide the basis for effective solutions in the future.

Pattern techniques are generally acknowledged to have been established as a valuable architectural design technique by Christopher Alexander, a buildings architect, who described this approach in his book *The Timeless Way of Building*, published in 1979. This book provides an introduction to the ideas behind the use of patterns, and Alexander followed it with two further books (*A Pattern Language* and *The Oregon Experiment*) in which he expanded on his description of the features and benefits of a patterns approach to architecture.

Software and buildings architects have many similar issues to address, and so it was natural for software architects to take an interest in patterns as an architectural tool. Many papers and books have been published on them since Alexander's 1979 book, perhaps the most renowned being *Design Patterns: Elements of Re-usable Object-Oriented Software* (Gamma et al., 1994). This book describes simple and elegant solutions to specific problems in object-oriented software design.

22.1.2 Content of a Pattern

Several different formats are used in the literature for describing patterns, and no single format has achieved widespread acceptance. However, there is broad agreement on the types of things that a pattern should contain. The headings which follow are taken from *Pattern-Oriented Software Architecture: A System of Patterns* (Buschmann et al., 1996). The elements described below will be found in most patterns, even if different headings are used to describe them.

Name — A meaningful and memorable way to refer to the pattern, typically a single word or short phrase.

Problem — A description of the problem indicating the intent in applying the pattern — the intended goals and objectives to be reached within the context and forces described below (perhaps with some indication of their priorities).

Context — The preconditions under which the pattern is applicable — a description of the initial state before the pattern is applied.

Forces — A description of the relevant forces and constraints, and how they interact/conflict with each other and with the intended goals and objectives. The description should clarify the intricacies of the problem and make explicit the kinds of trade-offs that must be considered. (The need for such trade-offs is typically what makes the problem difficult, and generates the need for the pattern in the first place.) The notion of "forces" equates in many ways to the "qualities" that architects seek to optimize, and the concerns they seek to address, in designing architectures. For example:

- Security, robustness, reliability, fault-tolerance
- Manageability
- Efficiency, performance, throughput, bandwidth requirements, space utilization
- Scalability (incremental growth on-demand)
- Extensibility, evolvability, maintainability
- Modularity, independence, re-usability, openness, composability (plug-and-play), portability
- Completeness and correctness
- Ease-of-construction
- Ease-of-use
- etc., . . .

Solution — A description, using text and/or graphics, of how to achieve the intended goals and objectives. The description should identify both the solution's static structure and its dynamic behavior — the people and computing actors, and their collaborations. The description may include guidelines for implementing the solution. Variants or specializations of the solution may also be described.

Resulting Context — The post-conditions after the pattern has been applied. Implementing the solution normally requires trade-offs among competing forces.

This element describes which forces have been resolved and how, and which remain unresolved. It may also indicate other patterns that may be applicable in the new context. (A pattern may be one step in accomplishing some larger goal.)

Any such other patterns will be described in detail under Related Patterns.

Examples One or more sample applications of the pattern which illustrate each of the other elements: a specific problem, context, and set of forces; how the pattern is applied; and the resulting context.

Rationale An explanation/justification of the pattern as a whole, or of individual components within it, indicating how the pattern actually works, and why — how it resolves the forces to achieve the desired goals and objectives, and why this is "good". The Solution element of a pattern describes the external structure and behavior of the solution: the Rationale provides insight into its internal workings.

Related Patterns
The relationships between this pattern and others. These may be predecessor patterns, whose resulting contexts correspond to the initial context of this one; or successor patterns, whose initial contexts correspond to the resulting context of this one; or alternative patterns, which describe a different solution to the same problem, but under different forces; or co-dependent patterns, which may/must be applied along with this pattern.

Known Uses Known applications of the pattern within existing systems, verifying that the pattern does indeed describe a proven solution to a recurring problem. Known Uses can also serve as Examples.

Patterns may also begin with an Abstract providing an overview of the pattern and indicating the types of problems it addresses. The Abstract may also identify the target audience and what assumptions are made of the reader.

22.1.3 Terminology

Although design patterns have been the focus of widespread interest in the software industry for several years, particularly in the object-oriented and component-based software fields, it is only recently that there has been increasing interest in architecture patterns — extending the principles and concepts of design patterns to the architecture domain.

The technical literature relating to this field is complicated by the fact that many people in the software field use the term "architecture" to refer to software, and many patterns described as "architecture patterns" are high-level software design patterns. This simply makes it all the more important to be precise in the use of terminology.

22.1.3.1 Architecture Patterns and Design Patterns

The term "design pattern" is often used to refer to any pattern which addresses issues of software architecture, design, or programming implementation. In *Pattern-Oriented Software Architecture: A System of Patterns*, the authors define these three types of patterns as follows:

- An **Architecture Pattern** expresses a fundamental structural organization or schema for software systems

  It provides a set of predefined subsystems, specifies their responsibilities, and includes rules and guidelines for organizing the relationships between them.

- A **Design Pattern** provides a scheme for refining the subsystems or components of a software system, or the relationships between them

  It describes a commonly recurring structure of communicating components that solves a general design problem within a particular context.

- An **Idiom** is a low-level pattern specific to a programming language

  An idiom describes how to implement particular aspects of components or the relationships between them using the features of the given language.

These distinctions are useful, but it is important to note that architecture patterns in this context still refers solely to software architecture. Software architecture is certainly an important part of the focus of the TOGAF standard, but it is not its only focus.

In this section we are concerned with patterns for enterprise system architecting. These are analogous to software architecture and design patterns, and borrow many of their concepts and terminology, but focus on providing re-usable models and methods specifically for the architecting of enterprise information systems — comprising software, hardware, networks, and people — as opposed to purely software systems.

22.1.3.2 Patterns and the Architecture Continuum

Although architecture patterns have not (as yet) been integrated into the TOGAF standard, each of the first four main phases of the ADM (Phases A through D) gives an indication of the stage at which relevant re-usable architecture assets from the Enterprise Architecture Continuum should be considered for use. Architecture patterns are one such asset.

An enterprise that adopts a formal approach to the use and re-use of architecture patterns will normally integrate their use into the Enterprise Architecture Continuum.

22.1.3.3 Patterns and Views

Architecture views are selected parts of one or more models representing a complete system architecture, focusing on those aspects that address the concerns of one or more stakeholders. Patterns can provide help in designing such models, and in composing views based on them.

22.1.3.4 Patterns and Business Scenarios

Relevant architecture patterns may well be identified in the work on business scenarios.

22.2 Some Pattern Resources

- The Patterns Home Page (refer to hillside.net/patterns) hosted by the Hillside Group provides information about patterns, links to online patterns, papers, and books dealing with patterns, and patterns-related mailing lists
- The Patterns-Discussion FAQ (refer to g.oswego.edu/dl/pd-FAQ/pd-FAQ.html) maintained by Doug Lea provides a very thorough and highly readable FAQ about patterns
- *Patterns and Software: Essential Concepts and Terminology* by Brad Appleton (refer to www.bradapp.com/docs/patterns-intro.html) provides another thorough and readable account of the patterns field

- The SOA Patterns community website (refer to www.soapatterns.org/), dedicated to the ongoing development and expansion of the SOA design pattern catalog
- The Cloud Computing Design Patterns community website (refer to www.cloudpatterns.org)

Chapter 23

Gap Analysis

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

23.1 Introduction

A key step in validating an architecture is to consider what may have been forgotten. The architecture must support all of the essential information processing needs of the organization. The most critical source of gaps that should be considered is stakeholder concerns that have not been addressed in prior architectural work.

Potential sources of gaps include:

- Business domain gaps:
  - People gaps (e.g., cross-training requirements)
  - Process gaps (e.g., process inefficiencies)
  - Tools gaps (e.g., duplicate or missing tool functionality)
  - Information gaps
  - Measurement gaps
  - Financial gaps
  - Facilities gaps (buildings, office space, etc.)
- Data domain gaps:
  - Data not of sufficient currency
  - Data not located where it is needed
  - Not the data that is needed
  - Data not available when needed
  - Data not created
  - Data not consumed
  - Data relationship gaps

- Applications impacted, eliminated, or created
- Technologies impacted, eliminated, or created

23.2 Suggested Steps

The suggested steps are as follows:

- Draw up a matrix with all the Architecture Building Blocks (ABBs) of the Baseline Architecture on the vertical axis, and all the ABBs of the Target Architecture on the horizontal axis
- Add to the Baseline Architecture axis a final row labeled "New", and to the Target Architecture axis a final column labeled "Eliminated"
- Where an ABB is available in both the Baseline and Target Architectures, record this with "Included" at the intersecting cell
- Where an ABB from the Baseline Architecture is missing in the Target Architecture, each must be reviewed

  If it was correctly eliminated, mark it as such in the appropriate "Eliminated" cell. If it was not, an accidental omission in the Target Architecture has been uncovered that must be addressed by reinstating the ABB in the next iteration of the architecture design — mark it as such in the appropriate "Eliminated" cell.
- Where an ABB from the Target Architecture cannot be found in the Baseline Architecture, mark it at the intersection with the "New" row as a gap that needs to filled, either by developing or procuring the building block

When the exercise is complete, anything under "Eliminated" or "New" is a gap, which should either be explained as correctly eliminated, or marked as to be addressed by reinstating or developing/procuring the function.

23.3 Example

Figure 23-1 shows an example analysis for ABBs that are services from the Network Services category of the TOGAF TRM, and shows a number of services from the Baseline Architecture missing from the Target Architecture.

Target → Architecture / Baseline Architecture ↓	Video Conferencing Services	Enhanced Telephony Services	Mailing List Services	Eliminated Services ↓
Broadcast Services				Intentionally eliminated
Video Conferencing Services	Included			
Enhanced Telephony Services		Potential match		
Shared Screen Services				Unintentionally excluded - a gap in Target Architecture
New →		Gap: Enhanced services to be developed or produced	Gap: To be developed or produced	

Figure 23-1 Gap Analysis Example

Chapter 24

Migration Planning Techniques

This chapter contains a number of techniques used to support migration planning in Phases E and F.

24.1 Implementation Factor Assessment & Deduction Matrix

The technique of creating an Implementation Factor Assessment and Deduction matrix can be used to document factors impacting the architecture Implementation and Migration Plan.

The matrix should include a list of the factors to be considered, their descriptions, and the deductions that indicate the actions or constraints that have to be taken into consideration when formulating the plans.

Factors typically include:

- Risks
- Issues
- Assumptions
- Dependencies
- Actions
- Impacts

An example matrix is shown in Figure 24-1.

Implementation Factor Assessment and Deduction Matrix		
Factor	**Description**	**Deduction**
<Name of Factor>	<Description of Factor>	<Impact on Migration Plan>
Change in Technology	Shut down the message centers, saving 700 personnel, and have them replaced by email.	• Need for personnel training, re-assignment • Email has major personnel savings and should be given priority
Consolidation of Services		
Introduction of New Customer Service		

Figure 24-1 Implementation Factor Assessment and Deduction Matrix

24.2 Consolidated Gaps, Solutions, & Dependencies Matrix

The technique of creating a Consolidated Gaps, Solutions, and Dependencies matrix allows the architect to group the gaps identified in the domain architecture gap analysis results and assess potential solutions and dependencies to one or more gaps.

This matrix can be used as a planning tool when creating work packages. The identified dependencies will drive the creation of projects and migration planning in Phases E and F.

An example matrix is shown in Figure 24-2.

Consolidated Gaps, Solutions, and Dependencies Matrix				
No.	**Architecture**	**Gap**	**Potential Solutions**	**Dependencies**
1	Business	New Order Processing Process	Use COTS software tool process Implement custom solution	Drives applications (2)
2	Application	New Order Processing Application	COTS software tool X Develop in-house	
3	Information	Consolidated Customer Information Base	Use COTS customer base Develop customer data mart	

Figure 24-2 Consolidated Gaps, Solutions, and Dependencies Matrix

24.3 Architecture Definition Increments Table

The technique of creating an Architecture Definition Increments table allows the architect to plan a series of Transition Architectures outlining the status of the Enterprise Architecture at specified times.

A table should be drawn up, as shown in Figure 24-3, listing the projects and then assigning their incremental deliverables across the Transition Architectures.

Architecture Definition - Project Objectives by Increment (Example Only)				
	April 2018/2019	**April 2019/2020**	**April 2020/2021**	
Project	**Transition Architecture 1: Preparation**	**Transition Architecture 2: Initial Operational Capability**	**Transition Architecture 3: Benefits**	**Comments**
Enterprise e-Services Capability	Training and Business Process	e-Licensing Capability	e-Employment Benefits	
IT e-Forms	Design and Build			
IT e-Information Environment	Design and Build Information Environment	Client Common Data Web Content Design and Build	Enterprise Common Data Component Management Design and Build	
. . .	. . .	. . .	. . .	. . .

Figure 24-3 Architecture Definition Increments Table

24.4 Transition Architecture State Evolution Table

The technique of creating the Transition Architecture State Evolution table allows the architect to show the proposed state of the architectures at various levels using the defined taxonomy (e.g., the TOGAF TRM).

A table should be drawn, listing the services from the taxonomy used in the enterprise, the Transition Architectures, and proposed transformations, as shown in Figure 24-4.

All Solution Building Blocks (SBBs) should be described with respect to their delivery and impact on these services. They should also be marked to show the progression of the Enterprise Architecture. In the example, where target capability has been reached, this is shown as "new" or "retain"; where capability is transitioned to a new solution, this is marked as "transition"; and where a capability is to be replaced, this is marked as "replace".

Architectural State using the Technical Reference Model				
Sub-Domain	**Service**	**Transition Architecture 1**	**Transition Architecture 2**	**Transition Architecture 3**
Infrastructure Applications	Information Exchange Services	Solution System A (replace)	Solution System B-1 (transition)	Solution System B-2 (new)
	Data Management Services	Solution System D (retain)	Solution System D (retain)	Solution System D (retain)
. . .	. . .	. . .	. . .	. . .

Figure 24-4 Transition Architecture State Evolution Table

Another technique (not shown here) is to use color coding in the matrix; for example:

- Green: service SBB in place (either new or retained)
- Yellow: service being transitioned into a new solution
- Red: service to be replaced

24.5 Business Value Assessment Technique

A technique to assess business value is to draw up a matrix based on a value index dimension and a risk index dimension. An example is shown in Figure 24-5. The value index should include criteria such as compliance to principles, financial contribution, strategic alignment, and competitive position. The risk index should include criteria such as size and complexity, technology, organizational capacity, and impact of a failure. Each criterion should be assigned an individual weight.

The index and its criteria and weighting should be developed and approved by senior management. It is important to establish the decision-making criteria before the options are known.

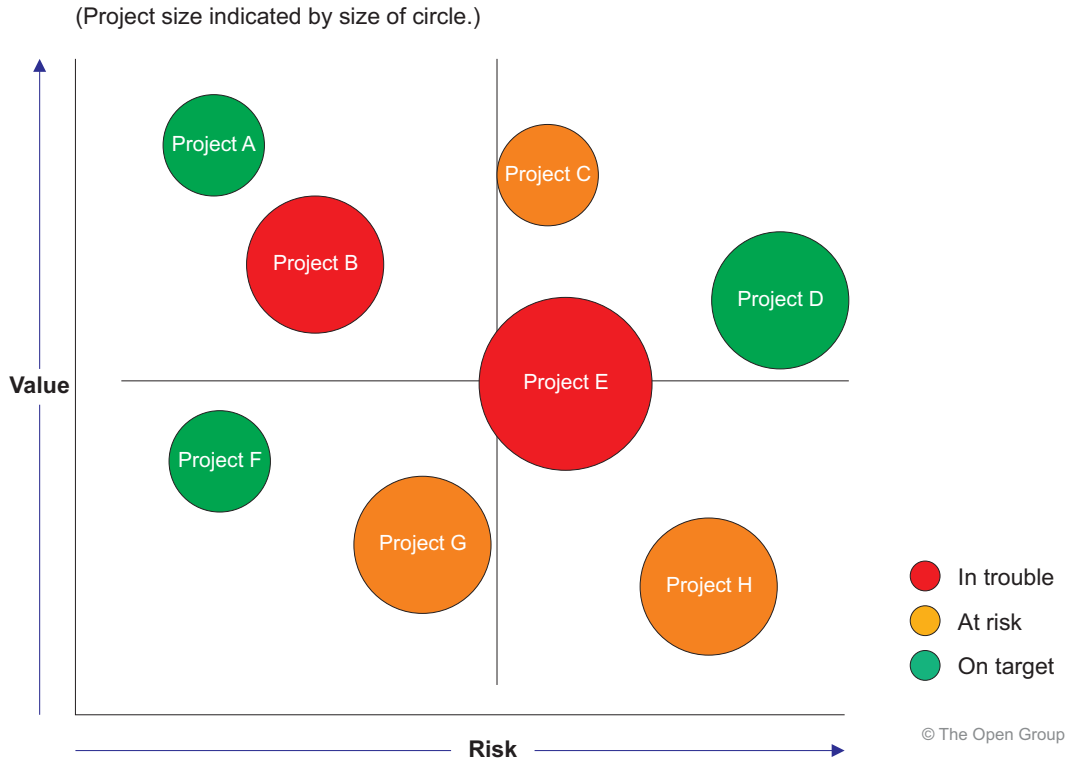

Figure 24-5 Sample Project Assessment with Respect to Business Value and Risk

Chapter 25

Interoperability Requirements

This chapter provides guidelines for defining and establishing interoperability requirements.

25.1 Overview

A definition of interoperability is "the ability to share information and services". Defining the degree to which the information and services are to be shared is a very useful architectural requirement, especially in a complex organization and/or extended enterprise.

The determination of interoperability is present throughout the Architecture Development Method (ADM) as follows:

- In the Architecture Vision (Phase A), the nature and security considerations of the information and service exchanges are first revealed within the business scenarios
- In the Business Architecture (Phase B), the information and service exchanges are further defined in business terms
- In the Data Architecture (Phase C), the content of the information exchanges is detailed using the corporate data and/or information exchange model
- In the Application Architecture (Phase C), the way that the various applications are to share the information and services is specified
- In the Technology Architecture (Phase D), the appropriate technical mechanisms to permit the information and service exchanges are specified
- In Opportunities & Solutions (Phase E), the actual solutions (e.g., Commercial Off-The-Shelf (COTS) packages) are selected
- In Migration Planning (Phase F), the interoperability is logically implemented

25.2 Defining Interoperability

There are many ways to define interoperability and the aim is to define one that is consistently applied within the enterprise and extended enterprise. It is best that both the enterprise and the extended enterprise use the same definitions.

Many organizations find it useful to categorize interoperability as follows:

- **Operational or Business Interoperability** defines how business processes are to be shared
- **Information Interoperability** defines how information is to be shared
- **Technical Interoperability** defines how technical services are to be shared or at least connect to one another

From an IT perspective, it is also useful to consider interoperability in a similar vein to Enterprise Application Integration (EAI); specifically:

- **Presentation Integration/Interoperability** is where a common look-and-feel approach through a common portal-like solution guides the user to the underlying functionality of the set of systems
- **Information Integration/Interoperability** is where the corporate information is seamlessly shared between the various corporate applications to achieve, for example, a common set of client information

  Normally this is based upon a commonly accepted corporate ontology and shared services for the structure, quality, access, and security/privacy for the information.
- **Application Integration/Interoperability** is where the corporate functionality is integrated and shareable so that the applications are not duplicated (e.g., one change of address service/component; not one for every application) and are seamlessly linked together through functionality such as workflow

  This impacts the business and infrastructure applications and is very closely linked to corporate business process unification/interoperability.
- **Technical Integration/Interoperability** includes common methods and shared services for the communication, storage, processing, and access to data primarily in the application platform and communications infrastructure domains

  This interoperability is premised upon the degree of rationalization of the corporate IT infrastructure, based upon standards and/or common IT platforms. For example, multiple applications sharing one infrastructure or 10,000 corporate websites using one centralized content management/web server (rather than thousands of servers and webmasters spread throughout the country/globe).

Many organizations create their own interoperability models, such as illustrated in the example below from the Canadian Government. They have a high-level definition of the three classes of interoperability and identify the nature of the information and services that they wish to share. Interoperability is coined in terms of e-enablers for e-Government. Their interoperability breakdown is as follows:

- Information Interoperability:
  - Knowledge management
  - Business intelligence
  - Information management
  - Trusted identity
- Business Interoperability:
  - Delivery networks
  - e-Democracy
  - e-Business
  - Enterprise resource management
  - Relationship and case management

- Technical Interoperability:
  - IT infrastructure

In certain architectural approaches, such as system of systems or a federated model, interoperability is a strongly recommended best practice that will determine how the systems interact with each other. A key consideration will be the enterprise's business operating model.

25.3 Enterprise Operating Model

Key to establishing interoperability is the determination of the corporate operating model, where the operating model is "the necessary level of business process integration and standardization for delivering goods and services to customers. An operating model describes how a company wants to thrive and grow. By providing a more stable and actionable view of the company than strategy, the operating model drives the design of the foundation for execution."[4]

For example, if lines of business or business units only need to share documents, then the Architecture and Solution Building Blocks (ABBs and SBBs) may be simpler than if there is a need to share structured transaction data. Similarly, if the Architecture Vision includes a shared services environment, then it is useful to define the level the services are to be shared.

The corporate operating model will normally indicate what type of interoperability approach will be appropriate. This model should be determined in Phase A (Architecture Vision) if not in Phase B (Business Architecture), and definitely by Phase E (Opportunities & Solutions).

Complex enterprises and/or extended enterprises (e.g., supply chain) may have more than one type of operating model. For example, it is common for the internal operating model (and supporting interoperability model) to differ from the one used for the extended enterprise.

25.4 Refining Interoperability

Implementing interoperability requires the creation, management, acceptance, and enforcement of realistic standards that are SMART (Specific, Measurable, Actionable, Realistic, and Time-bound). Clear measures of interoperability are key to success.

Architecture is the key for identifying standards and facilitated sessions (brainstorming) will examine potential pragmatic ways (that fit within the current or emerging business culture) to achieve the requisite degree of interoperability.

Interoperability should be refined so that it meets the needs of the enterprise and/or extended enterprise in an unambiguous way. The refined interoperability measures (degrees, types, and high-level targets) should be part of or referred to the Enterprise Architecture strategic direction.

These measures are instantiated within a transformation strategy that should be embedded within the Target Architecture definition and pragmatically implemented in the Transition Architectures. Upon completion, also update the consolidated gap analysis results and dependencies to ensure that all of the brainstorming nuggets are captured.

An example of specifying interoperability is the Degrees of Interoperability (used within the Canadian Department of National Defense and NATO). These organizations were focused on the sharing of information and came up with four degrees of interoperability as follows:

---

4. *Enterprise Architecture as Strategy* (Ross et al., 2006) provides potential models.

- **Degree 1: Unstructured Data Exchange** involves the exchange of human-interpretable unstructured data, such as the free text found in operational estimates, analysis, and papers
- **Degree 2: Structured Data Exchange** involves the exchange of human-interpretable structured data intended for manual and/or automated handling, but requires manual compilation, receipt, and/or message dispatch
- **Degree 3: Seamless Sharing of Data** involves the automated sharing of data amongst systems based on a common exchange model
- **Degree 4: Seamless Sharing of Information** is an extension of Degree 3 to the universal interpretation of information through data processing based on co-operating applications

These degrees should be further refined and made technically meaningful for each of the degrees. An example refinement of Degree 3 with four subclassifications follows:

- 3A: Formal Message Exchange
- 3B: Common Data Exchange
- 3C: Complete Data Exchange
- 3D: Real-time Data Exchange

The intent is to specify the detailed degrees of interoperability to the requisite level of detail so that they are technically meaningful.

These degrees are very useful for specifying the way that information has to be exchanged between the various systems and provide critical direction to the projects implementing the systems.

Similar measures should be established to determine service/business and technical interoperability.

25.5 Determining Interoperability Requirements

Co-existence between emerging and existing systems, especially during transformation, will be a major challenge and brainstorming should attempt to figure out what has to be done to reduce the pain. It is imperative to involve the operations management staff and architects in this step as they will be responsible for operating the portfolio deliverables.

For example, there might be a need for a "wrapper" application (an application that acts as the interface [a.k.a. interpreter] between the legacy application and the emerging infrastructure). Indeed, pragmatically, in the "if it works do not fix it" world, the "wrapper" might become a permanent solution.

Regardless, using the gap analysis results and business scenarios as a foundation, brainstorm the IT issues and work them through to ensure that all of the gaps are clearly identified and addressed and verify that the organization-specific requirements will be met.

It is important to note that the ensuing development process must include recognition of dependencies and boundaries for functions and should take account of what products are available in the marketplace. An example of how this might be expressed can be seen in the building blocks example (see Part III, Chapter 33).

If a mechanism such as the Degrees of Interoperability is used, then a matrix showing the interoperability requirements is a useful tool, as illustrated in Figure 25-1 and Figure 25-2, noting that the degree of information sharing is not necessarily symmetrical or bidirectional between systems and/or stakeholders.

The matrix below can be used within the enterprise and/or within the extended enterprise as a way of detailing that information and/or services can be shared. The matrix should start in the Business Architecture (Phase B) to capture the nature of the sharing of information between stakeholders, and evolve to determine the what systems share what information in Phase C.

Phase B: Inter-stakeholder Information Interoperability Requirements (Using degrees of information interoperability)							
Stakeholders	A	B	C	D	E	F	G
A		2	3	2	3	3	3
B	2		3	2	3	2	2
C	3	3		2	2	2	3
D	2	2	2		3	3	3
E	4	4	2	3		3	3
F	4	4	2	3	3		2
G	2	2	3	3	3	3	

Figure 25-1 Business Information Interoperability Matrix

Figure 25-1 shows that Stakeholder A requires structured data exchange (Degree 2) with Stakeholders/Systems B and D, and seamless sharing of data (Degree 3) with Stakeholders/Systems C, E, F, and G.

The business information interoperability matrix should be refined within the Information Systems Architecture using refined measures and specifying the actual systems used by the stakeholders. A sample is shown in Figure 25-2.

Phase C: Inter-system Interoperability Requirements							
	System A	System B	System C	System D	System E	System F	System G
System A		2A	3D	2B	3A	3A	3B
System B	2E		3F	2C	3A	2B	2C
System C	3E	3F		2B	2A	2A	3B
System D	2B	2B	2B		3A	3A	3B
System E	4A	4B	2B	3A		3B	3B
System F	4A	4A	2B	3B	3A		2D
System G	2B	2B	3A	3A	3B	3B	

Figure 25-2 Information Systems Interoperability Matrix

In Figure 25-2, both the nature of the exchange is more detailed (e.g., Degree 3A *versus* only Degree 3) and the sharing is between specific systems rather than stakeholders. For example, System A shares information with the other systems in accordance with enterprise technical standards.

In many organizations the Business Architectures describe the nature of the information shared

between stakeholders and/or organizations (e.g., in defense the term is "operational node"), and the Data Architecture specifies the information shared between systems.

Update the defined target data and Application Architecture (Version 1.0) with the interoperability issues that were raised.

25.6 Reconciling Interoperability Requirements with Potential Solutions

The Enterprise Architect will have to ensure that there are no interoperability conflicts, especially if there is an intention to re-use existing SBBs and/or COTS.

The most significant issue to be addressed is in fact business interoperability. Most SBBs or COTS will have their own business processes embedded. Changing the embedded business processes will often require so much work that the advantages of re-using solutions will be lost. There are numerous examples of this in the past.

Furthermore, there is the workflow aspect between the various systems that has to be taken into account. The Enterprise Architect will have to ensure that any change to the business interoperability requirements is signed off by the Business Architects and architecture sponsors in a revised Statement of Architecture Work.

Chapter 26

Business Transformation Readiness Assessment

This chapter describes a technique known as Business Transformation Readiness Assessment, used for evaluating and quantifying an organization's readiness to undergo change.

This chapter builds on work by the Canadian Government and its Business Transformation Enablement Program (BTEP).[5]

26.1 Introduction

Enterprise Architecture is a major endeavor within an organization and most often an innovative Architecture Vision (Phase A) and supporting Architecture Definition (Phases B to D) will entail considerable change. There are many dimensions to change, but by far the most important is the human element. For example, if the enterprise envisages a consolidation of information holdings and a move to a new paradigm such as service orientation for integrated service delivery, then the human resource implications are major. Potentially coupled with a change-averse culture and a narrowly skilled workforce, the most sound and innovative architecture could go nowhere.

Understanding the readiness of the organization to accept change, identifying the issues, and then dealing with them in the Implementation and Migration Plans is key to successful architecture transformation in Phases E and F. This will be a joint effort between corporate (especially human resources) staff, lines of business, and IT planners.

The recommended activities in an assessment of an organization's readiness to address business transformation are:

- Determine the readiness factors that will impact the organization
- Present the readiness factors using maturity models
- Assess the readiness factors, including determination of readiness factor ratings
- Assess the risks for each readiness factor and identify improvement actions to mitigate the risk
- Work these actions into Phase E and F Implementation and Migration Plan

5. Refer to www.tbs-sct.gc.ca/btep-pto/index_e.asp.

26.1.1 Business Transformation Enablement Program (BTEP)

The Canadian Government Business Transformation Enablement Program (BTEP) provides guidance on how to identify the business transformation-related issues.

The BTEP recommends that all projects conduct a transformation readiness assessment to at least uncover the business transformation issues. This assessment is based upon the determination and analysis/rating of a series of readiness factors. The outcome is a deeper understanding of the challenges and opportunities that could be presented in the course of the endeavor. Many of the challenges translate directly into risks that have to be addressed, monitored, and, if possible, mitigated.

The following sections describe Business Transformation Readiness Assessment using the BTEP method, including some lessons learned. Readers should keep in mind that most organizations will have their own unique set of factors and criteria, but most are similar.

26.2 Determine Readiness Factors

The first step is to determine what factors will impact on the business transformation associated with the migration from the Baseline to Target Architectures.

This can be best achieved through the conduct of a facilitated workshop with individuals from different parts of the organization. It is important that all perspectives are sought as the issues will be varied. In this workshop it is very useful to start off with a tentative list of factors that participants can re-use, reject, augment, or replace.

An example set of factors drawn from the BTEP follows:

- **Vision** is the ability to clearly define and communicate what is to be achieved

  This is where management is able to clearly define the objectives, in both strategic and specific terms. Leadership in defining vision and needs comes from the business side with IT input. Predictable and proven processes exist for moving from vision to statement of requirements. The primary drivers for the initiative are clear. The scope and approach of the transformation initiative have been clearly defined throughout the organization.

- **Desire, Willingness, and Resolve** is the presence of a desire to achieve the results, willingness to accept the impact of doing the work, and the resolve to follow through and complete the endeavor

  There is active discussion regarding the impact that executing the project may have on the organization, with clear indication of the intent to accept the impacts. Key resources (e.g., financial, human, etc.) are allocated for the endeavor and top executives project the clear message that the organization will follow through; a message that identifies the effort as well as the benefits. Organizationally there is a history of finishing what is started and of coming to closure on issues in the timeframes needed and there is agreement throughout the organization that the transformation initiative is the "right" thing to do.

- **Need**, in that there is a compelling need to execute the endeavor

  There are clear statements regarding what the organization will not be able to do if the project does not proceed, and equally clear statements of what the project will enable the organization to do. There are visible and broadly understood consequences of endeavor failure and success criteria have been clearly identified and communicated.

- **Business Case** exists that creates a strong focus for the project, identifying benefits that must be achieved and thereby creating an imperative to succeed

The business case document identifies concrete benefits (revenues or savings) that the organization is committed to deliver and clearly and unquestionably points to goals that the organization is committed to achieving.

- **Funding**, in the form of a clear source of fiscal resources, exists that meets the endeavor's potential expenditures
- **Sponsorship and Leadership** exists and is broadly shared, but not so broad as to diffuse accountability

  Leadership keeps everyone "on board" and keeps all focused on the strategic goals. The endeavor is sponsored by an executive who is appropriately aligned to provide the leadership the endeavor needs and able to articulate and defend the needs of the endeavor at the senior management level. These executive sponsors are and will remain engaged throughout.

- **Governance** is the ability to engage the involvement and support of all parties with an interest in or responsibility to the endeavor with the objective of ensuring that the corporate interests are served and the objectives achieved

  There are clearly identified stakeholders and a clear sense of their interest in and responsibility to the project; a culture that encourages participation towards corporate rather than local objectives; a history of being able to successfully manage activities that cross interest areas; a culture that fosters meaningful, as opposed to symbolic, participation in management processes; and a commitment to ongoing project review and challenge and openness to outside advice.

- **Accountability** is the assignment of specific and appropriate responsibility, recognition of measurable expectations by all concerned parties, and alignment of decision-making with areas of responsibility and with where the impact of the decisions will be felt

  Accountability is aligned with the area where the benefits of success or consequences of failure of the endeavor will be felt as well as with the responsibility areas.

- **Workable Approach and Execution Model** is an approach that makes sense relative to the task, with a supporting environment, modeled after a proven approach

  There are clear notions of the client and the client's role relative to the builder or prime contractor and the organization is experienced with endeavors of this type so that the processes, disciplines, expertise, and governance are already in place, proven, and available to apply to the transformation endeavor. All the players know their roles because they have played them before with success. In particular, the roles of "client" and "systems builder" are mature and stable. There is a communication plan covering all levels of the organization and meeting the needs ranging from awareness to availability of technical detail. There is a reward and recognition plan in place to recognize teams and individuals who use good change management practices, planning and prevention of crisis behaviors, and who reinforce behaviors appropriate to the new way of doing business. It is clear to everyone how implementation will occur, how it will be monitored, and how realignment actions will be made and there are adequate resources dedicated for the life of the transformation.

- **IT Capacity to Execute** is the ability to perform all the IT tasks required by the project, including the skills, tools, processes, and management capability

  There has been a recent successful execution of a similar endeavor of similar size and complexity and there exist appropriate processes, discipline, skills, and a rationale model for deciding what skills and activities to source externally.

- **Enterprise Capacity to Execute** is the ability of the enterprise to perform all the tasks required by the endeavor, in areas outside of IT, including the ability to make decisions within the tight time constraints typical to project environments based upon the recent successful execution of a similar endeavor of at least half the size and complexity

  There exist non-IT-specific processes, discipline, and skills to deal with this type of endeavor. The enterprise has a demonstrated ability to deal with the type of ongoing project/portfolio management issues and requirements. There is a recognition of the need for knowledge and skill-building for the new way of working as well as the value of a formal gap analysis for skills and behavior.

- **Enterprise Ability to Implement and Operate** the transformation elements and their related business processes, absorb the changes arising from implementation, and ongoing ability to operate in the new environment

  The enterprise has a recent proven ability to deal with the change management issues arising from new processes and systems and has in place a solid disciplined and process-driven service management program that provides operations, maintenance, and support for existing systems.

Once the factors have been identified and defined, it is useful to call a follow-on workshop where the factors shall be assessed in some detail in terms of their impact/risk. The next section will deal with preparing for an effective assessment of these factors.

26.3 Present Readiness Factors

Once the factors are determined, it is necessary to present them in such a way that the assessment is clear and the maximum value is derived from the participants.

One such presentation is through the use of maturity models. If each factor is converted into a maturity model (a re-usable governance asset as well) accompanied by a standard worksheet template containing all of the information and deductions that have to be gathered, it can be a very useful tool.

The maturity model should enable participants to:

- Assess their current (Baseline Architecture) maturity level
- Determine the target maturity level that would have to be achieved to realize the Target Architecture
- Determine an intermediate target that would be achievable in a lesser timeframe

The care spent preparing the models (which is not insignificant) will be recouped by a focused workshop that will rapidly go through a significant number of factors.

It is important that each factor be well-defined and that the scope of the Enterprise Architecture endeavor (preliminary planning) be reflected in the models to keep the workshop participants focused and productive.

Circulating the models before the workshop for comments would be useful, if only to ensure that they are complete as well as allowing the participants to prepare for the workshop. Note that the model shown below also has a recommended target state put in by the Enterprise Architect; this again acts as governance.

An example of a maturity model is shown in Figure 26-1 for one of the BTEP factors.

Business Transformation Readiness Assessment - Maturity Model					
Factor 2: Need for Enterprise Information Architecture			**Class**	Organizational Context	
			BTEP Readiness Factor	YES	
Definition	There is recognition by the organization that information is a strategic corporate asset requiring stewardship. There is also recognition that the data is not universally understandable, of requisite quality, and accessible.				
Maturity Model Levels					
0 Not defined	**1 *Ad Hoc***	**2 Repeatable**	**3 Defined**	**4 Managed**	**5 Optimized**
Information is not recognized as an asset. There is no clear stewardship of data.	Data Management (DM) concepts are intuitively understood and practiced on an *ad hoc* basis. Stewardship of the data is informal. Data is recognized by certain internal experts and senior management as being of strategic importance to the organization. Focus is primarily on technically managing redundant data at the applications level.	Many parts of the organization value information/data as a strategic asset. Internal DM experts maintain clear lines of responsibility and stewardship of the data, organized along lines of business and at all senior levels. Staff put into practice DM principles and standards in their daily activities.	Data is recognized as a strategic asset in most parts of the organization, and throughout most levels from operations to senior management. Resources are committed to ensuring strong stewardship of data at the lower management and information expert levels.	Data is recognized as a strategic asset in all parts of the organization, and throughout most levels from operations to senior management. Resources are committed to ensuring strong stewardship of data at the senior management and information expert levels.	Data is treated in all levels throughout the organization as a strategic asset to be exploited and re-used. Data products and services are strongly integrated with the management practice of the organization. All staff are empowered and equipped to take stewardship of information, and are seen as "knowledge workers".
				Recommended Target State	

Figure 26-1 Business Transformation Readiness Assessment — Maturity Model

26.4 Assess Readiness Factors

Ideally, the factors should be assessed in a multi-disciplinary workshop. Using a mechanism such as maturity models, Enterprise Architects will normally have to cover a great deal of ground in little time.

The use of a series of templates for each factor would expedite the assessment, and ensure consistency across the wide range of factors.

The assessment should address three things, namely:

- Readiness Factor Vision
- Readiness Factor Rating
- Readiness Factor Risks & Actions

26.4.1 Readiness Factor Vision

The vision for a readiness factor is the determination of where the enterprise has to evolve to address the factor. First, the factor should be assessed with respect to its base state and then its target state.

For example, if the "IT capacity to execute" factor is rated as low, the factor should ideally be at "high" to realize the Target Architecture Vision. An intermediate target might be useful to direct the implementation. Maturity models are excellent vehicles to guide this determination.

26.4.2 Readiness Factor Rating

Once the factor visions are established, then it is useful to determine how important each factor is to the achievement of the Target Architecture as well as how challenging it will be to migrate the factor into an acceptable visionary state.

The BTEP uses a Readiness Rating Scheme that can be used as a start point for any organization in any vertical. Each one of the readiness factors are rated with respect to:

- **Urgency**, whereby if a readiness factor is urgent, it means that action is needed before a transformation initiative can begin
- **Readiness Status**, which is rated as either Low (needs substantial work before proceeding), Fair (needs some work before proceeding), Acceptable (some readiness issues exist; no showstoppers), Good (relatively minor issues exist), or High (no readiness issues)
- **Degree of Difficulty to Fix** rates the effort required to overcome any issues identified as either No Action Needed, Easy, Moderate, or Difficult

Although a more extensive template can be used in the workshop, it is useful to create a summary table of the findings to consolidate the factors and provide a management overview. A like summary is shown in Figure 26-2.

Business Factor Assessment Summary				
Ser	**Readiness Factor**	**Urgency**	**Readiness Status**	**Degree of Difficulty to Fix**
1	Vision			
2	Desire/willingness/resolve			
3	Need			
4	Business case			
5	Funding			
6	Sponsorship and leadership			
7	Governance			
8	Accountability			
9	Workable approach and execution model			
10	IT capacity to execute			
11	Departmental capacity to execute			
12	Ability to implement and operate			

Figure 26-2 Summary Table of Business Transformation Readiness Assessment

26.4.3 Readiness Factor Risks & Actions

Once the factors have been rated and assessed, derive a series of actions that will enable the factors to change to a favorable state.

Each factor should be assessed with respect to risk using the process highlighted in Part III, Chapter 27, including an estimate of impact and frequency.

Each factor should be discretely assessed and a series of improvement actions outlined. Before starting anew, existing actions outlined in the architectures should be checked first before creating new ones.

These newly identified actions should then be formally incorporated into the emerging Implementation and Migration Plan.

From a risk perspective, these actions are designed to mitigate the risks and produce an acceptable residual risk. As risks, they should be part of the risk management process and closely monitored as the Enterprise Architecture is being implemented.

26.5 Readiness and Migration Planning

The assessment exercise will provide a realistic assessment of the organization and will be a key input into the strategic migration planning that will be initiated in Phase E and completed in Phase F. It is important to note whether the business transformation actions will be on the vision's critical path and, if so, determine how they will impact implementation. There is no point deploying new IT capability without employees trained to use it and support staff ready to sustain it.

The readiness factors, as part of an overall Implementation and Migration Plan, will have to be continuously monitored (Phase G) and rapid corrective actions taken through the IT governance framework to ensure that the defined architectures can be implemented.

The readiness factors assessment will be a living document and during the migration planning and execution of the Transition Architectures, the business transformation activities will play a key role.

26.6 Marketing the Implementation Plan

The Architecture Definition should not be widely circulated until the business transformation issues are identified and mitigated, and the associated actions part of an overall "marketing" plan for the vision and the Implementation and Migration Plan.

For example, the consolidation of information holdings could result in hundreds of lost jobs and this vision should not be announced before a supporting business transformation/human resources plan is formulated to retrain or support the workers' quest for new employment.

The business transformation workshops are a critical part of the Communications Plan whereby key individuals from within the organization gather to assess the implications of transforming the enterprise. To do this they will become aware of the Architecture Vision and architecture definition (if they were not already involved through the business scenarios and Business Architecture). This group will feel ownership of the Enterprise Architecture, recognizing the Enterprise Architect as a valuable steward.

Their determination of the factors will again create a culture of understanding across the enterprise and provide useful insights for the Implementation and Migration Plan.

The latter plan should include a Communications Plan, especially to keep the affected personnel informed. In many cases collaborating with the unions and shop stewards will further assist a humane (and peaceful) transition to the target state.

26.7 Conclusion

In short, Enterprise Architecture implementation will require a deep knowledge and awareness of all of the business transformation factors that impact transitioning to the visionary state. With the evolution of IT, the actual technology is not the real issue any more in Enterprise Architecture, but the critical factors are most often the cultural ones. Any Implementation and Migration Plan has to take both into consideration. Neglecting these and focusing on the technical aspects will invariably result in an implementation that falls short of realizing the real promise of a visionary Enterprise Architecture.

Chapter 27

Risk Management

This chapter describes risk management, which is a technique used to mitigate risk when implementing an architecture project.

27.1 Introduction

There will always be risk with any architecture/business transformation effort. It is important to identify, classify, and mitigate these risks before starting so that they can be tracked throughout the transformation effort.

Mitigation is an ongoing effort and often the risk triggers may be outside the scope of the transformation planners (e.g., merger, acquisition) so planners must monitor the transformation context constantly.

It is also important to note that the Enterprise Architect may identify the risks and mitigate certain ones, but it is within the governance framework that risks have to be first accepted and then managed.

There are two levels of risk that should be considered, namely:

1. **Initial Level of Risk**: risk categorization prior to determining and implementing mitigating actions
2. **Residual Level of Risk**: risk categorization after implementation of mitigating actions (if any)

The process for risk management is described in the following sections and consists of the following activities:

- Risk classification
- Risk identification
- Initial risk assessment
- Risk mitigation and residual risk assessment
- Risk monitoring

27.2 Risk Classification

Risk is pervasive in any Enterprise Architecture activity and is present in all phases within the Architecture Development Method (ADM). From a management perspective, it is useful to classify the risks so that the mitigation of the risks can be executed as expeditiously as possible.

One common way for risks to be classified is with respect to impact on the organization (as discussed in Section 27.4), whereby risks with certain impacts have to be addressed by certain levels of governance.

Risks are normally classified as time (schedule), cost (budget), and scope but they could also include client transformation relationship risks, contractual risks, technological risks, scope and complexity risks, environmental (corporate) risks, personnel risks, and client acceptance risks.

Another way of delegating risk management is to further classify risks by architecture domains. Classifying risks as business, information, applications, and technology is useful but there may be organizationally-specific ways of expressing risk that the corporate Enterprise Architecture directorate should adopt or extend rather than modify.

Ultimately, Enterprise Architecture risks are corporate risks and should be classified and as appropriate managed in the same or extended way.

27.3 Risk Identification

The maturity and transformation readiness assessments will generate a great many risks. Identify the risks and then determine the strategy to address them throughout the transformation.

The use of Capability Maturity Models (CMMs) is suitable for specific factors associated with architecture delivery to first identify baseline and target states and then identify the actions required to move to the target state. The implications of *not* achieving the target state can result in the discovery of risks. Refer to Chapter 26 for specific details.

Risk documentation is completed in the context of a Risk Management Plan, for which templates exist in standard project management methodologies — e.g., Project Management Book of Knowledge (PMBOK) and PRINCE2 — as well as with the various government methodologies.

Normally these methodologies involve procedures for contingency planning, tracking and evaluating levels of risk, reacting to changing risk level factors, as well as processes for documenting, reporting, and communicating risks to stakeholders.

27.4 Initial Risk Assessment

The next step is to classify risks with respect to effect and frequency in accordance with scales used within the organization. Combine effect and frequency to come up with a preliminary risk assessment.

There are no hard and fast rules with respect to measuring effect and frequency. The following guidelines are based upon existing risk management best practices. Effect could be assessed using the following example criteria:

- **Catastrophic** infers critical financial loss that could result in bankruptcy of the organization

- **Critical** infers serious financial loss in more than one line of business leading to a loss in productivity and no return on investment on the IT investment
- **Marginal** infers a minor financial loss in a line of business and a reduced return on investment on the IT investment
- **Negligible** infers a minimal impact on a line of business' ability to deliver services and/or products

Frequency could be indicated as follows:

- **Frequent**: likely to occur very often and/or continuously
- **Likely**: occurs several times over the course of a transformation cycle
- **Occasional**: occurs sporadically
- **Seldom**: remotely possible and would probably occur not more than once in the course of a transformation cycle
- **Unlikely**: will probably not occur during the course of a transformation cycle

Combining the two factors to infer impact would be conducted using a heuristically-based but consistent classification scheme for the risks. A potential scheme to assess corporate impact could be as follows:

- **Extremely High Risk (E)**: the transformation effort will most likely fail with severe consequences
- **High Risk (H)**: significant failure of parts of the transformation effort resulting in certain goals not being achieved
- **Moderate Risk (M)**: noticeable failure of parts of the transformation effort threatening the success of certain goals
- **Low Risk (L)**: certain goals will not be wholly successful

These impacts can be derived using a classification scheme, as shown in Figure 27-1.

Corporate Risk Impact Assessment					
Effect	Frequency				
	Frequent	Likely	Occasional	Seldom	Unlikely
Catastrophic	E	E	H	H	M
Critical	E	H	H	M	L
Marginal	H	M	M	L	L
Negligible	M	L	L	L	L

Figure 27-1 Risk Classification Scheme

27.5 Risk Mitigation and Residual Risk Assessment

Risk mitigation refers to the identification, planning, and conduct of actions that will reduce the risk to an acceptable level.

The mitigation effort could be a simple monitoring and/or acceptance of the risk to a full-blown contingency plan calling for complete redundancy in a Business Continuity Plan (with all of the associated scope, cost, and time implications).

Due to the implications of this risk assessment, it has to be conducted in a pragmatic but systematic manner. With priority going to frequent high impact risks, each risk has to be mitigated in turn.

27.6 Conduct Residual Risk Assessment

Once the mitigation effort has been identified for each one of the risks, re-assess the effect and frequency and then recalculate the impacts and see whether the mitigation effort has really made an acceptable difference. The mitigation efforts will often be resource-intensive and a major outlay for little or no residual risk should be challenged.

Once the initial risk is mitigated, then the risk that remains is called the "residual risk". The key consideration is that the mitigating effort actually reduces the corporate impact and does not just move the risk to another similarly high quadrant. For example, changing the risk from frequent/catastrophic to frequent/critical still delivers an Extremely high risk. If this occurs, then the mitigation effort has to be re-considered.

The final deliverable should be a transformation risk assessment that could be structured as a worksheet, as shown in Figure 27-2.

Risk ID	Risk	Preliminary Risk			Mitigation	Residual Risk		
		Effect	Frequency	Impact		Effect	Frequency	Impact

Figure 27-2 Sample Risk Identification and Mitigation Assessment Worksheet

27.7 Risk Monitoring and Governance (Phase G)

The residual risks have to be approved by the IT governance framework and potentially in corporate governance where business acceptance of the residual risks is required.

Once the residual risks have been accepted, then the execution of the mitigating actions has to be carefully monitored to ensure that the enterprise is dealing with residual rather than initial risk.

The risk identification and mitigation assessment worksheets are maintained as governance artifacts and are kept up-to-date in Phase G (Implementation Governance) where risk monitoring is conducted.

Implementation governance can identify critical risks that are not being mitigated and might require another full or partial ADM cycle.

27.8 Summary

Risk management is an integral part of Enterprise Architecture. Practitioners are encouraged to use their corporate risk management methodology or extend it using the guidance in this chapter. In the absence of a formal corporate methodology, architects can use the guidance in this chapter as a best practice.

Chapter 28

Capability-Based Planning

This chapter provides an overview of capability-based planning, a business planning technique that focuses on business outcomes. It also copes well with the friction of co-ordinating projects across corporate functional domains that together enable the enterprise to achieve that capability (for example, electronic service delivery).

28.1 Overview

Capability-based planning focuses on the planning, engineering, and delivery of strategic business capabilities to the enterprise. It is business-driven and business-led and combines the requisite efforts of all lines of business to achieve the desired capability. Capability-based planning accommodates most, if not all, of the corporate business models and is especially useful in organizations where a latent capability to respond (e.g., an emergency preparedness unit) is required and the same resources are involved in multiple capabilities. Often the need for these capabilities are discovered and refined using business scenarios (see the TOGAF® Series Guide: Business Scenarios).

From an IT perspective, capability-based planning is particularly relevant. For example, setting up a data center is really about consolidating corporate data and providing the related services. Lead Enterprise Architects for this capability will find themselves involved in managing construction, personnel training, and other change management tasks as well as IT architecture tasks. In the past, many IT projects were less than successful even though the actual IT implementation was brilliant, but the associated other tasks (business process re-engineering, client training, support training, infrastructure, and so on) were not controlled by the Enterprise Architects and planners and often were not satisfactorily completed.

On the other hand, IT projects were often described in terms of technical deliverables not as business outcomes, making it difficult for business to appreciate what was being delivered and often the IT architects lost sight of the ultimate business goal. Capability-based planning frames all phases of the architecture development in the context of business outcomes, clearly linking the IT vision, architectures (ABBs and SBBs), and the Implementation and Migration Plans with the corporate strategic, business, and line-of-business plans.

In many governments, horizontal interoperability and shared services are emerging as cornerstones of their e-Government implementations and capability-based management is also prominent although under many guises. In the private sector, the concepts of supply chain management and Service-Oriented Architecture (SOA) are increasingly forcing planners/managers to govern horizontally as well as vertically.

28.2 Capability-Based Planning Paradigm

Capability-based planning has long been entrenched in the Defense realm in the US, UK, Australia, and Canada. The associated governance mechanisms, as well as rigorous capability derivation (capability engineering), are emerging primarily in the systems engineering domain. These concepts are readily transferable into other domains, such as IT.

28.3 Concept of Capability-Based Planning

From an Enterprise Architecture and IT perspective, capability-based planning is a powerful mechanism to ensure that the strategic business plan drives the enterprise from a top-down approach. It is also adaptable with capability engineering to leverage emerging bottom-up innovations.

No matter how the corporation structures itself, it will have to cope with the delivery of business capabilities whose delivery will require co-ordination and alignment across business verticals.

Capabilities are business-driven and ideally business-led. One of the main challenges is that the benefits are often reaped at the enterprise and not the line-of-business level. Consequently, projects within line-of-business-led portfolios tend to take a line-of-business rather than corporate perspective. Managing the delivery of a capability is challenging, but the entrenchment of a capability-based perspective within an organization is a powerful mechanism to deliver synergistically derived business value that will resonate in profitability and stock value.

Capabilities should be specified using the same discipline in the specification of objectives as in business scenarios; specifically, they should follow the SMART guidelines to avoid ambiguity.

As shown in Figure 28-1, many capabilities are "horizontal" and go against the grain of normal vertical corporate governance. Most often, management direction as well as the corporate management accountability framework are based upon line of business metrics, not enterprise metrics. Enterprise Architecture is also a horizontal function that looks at enterprise-level (as well as line of business-level) optimization and service delivery. Not surprisingly, capability-based planning and Enterprise Architecture are mutually supportive. Both often operate against the corporate grain and both have to cope with challenging business environments. Business support of Enterprise Architecture is crucial for its success and it is logical that it aligns with the corporate capability planners as well as providing support for those within the vertical lines of business.

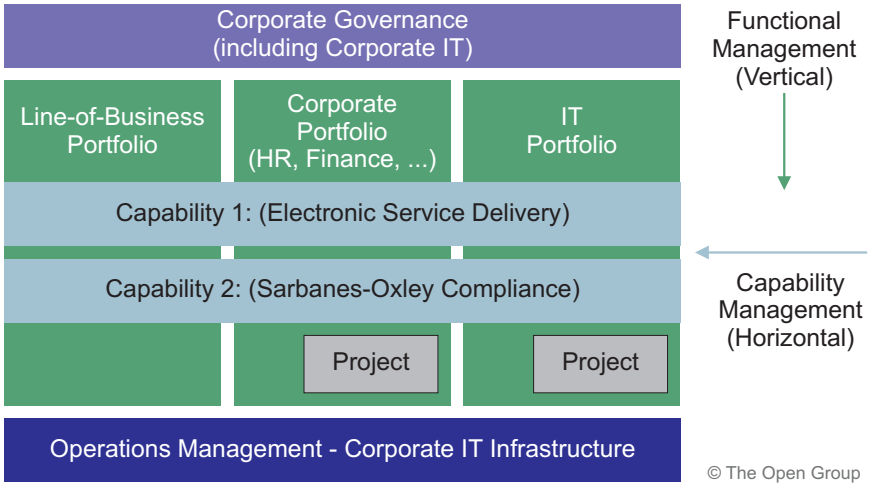

Figure 28-1 Capability-Based Planning Concept

Capabilities can also be vertical and handled in the context of the business organizational structure. In fact, capability requirements often drive organizational design, but within an organization in the process of business transformation, the organization may be trailing the capability needs.

Vertical capabilities are easier to handle and support by the Enterprise Architecture function, but still challenging when services are rationalized at the enterprise level and lines of business receive shared services that they do not directly control (they provide indirect control through IT governance in the Architecture Board as created in preliminary planning and used in Phase G (Implementation Governance).

For capability-based planning to succeed, it has to be managed with respect to dimensions and increments, as explained in the following two sections.

28.3.1 Capability Dimensions

Capabilities are engineered/generated taking into consideration various dimensions that straddle the corporate functional portfolios.

Every organization has a different but similar set of dimensions. An example set (based upon the Canadian Department of National Defense) could include personnel, research & development, infrastructure/facilities, concepts/processes, information management, and material. Whatever dimensions are selected, they should be well explained and understood.

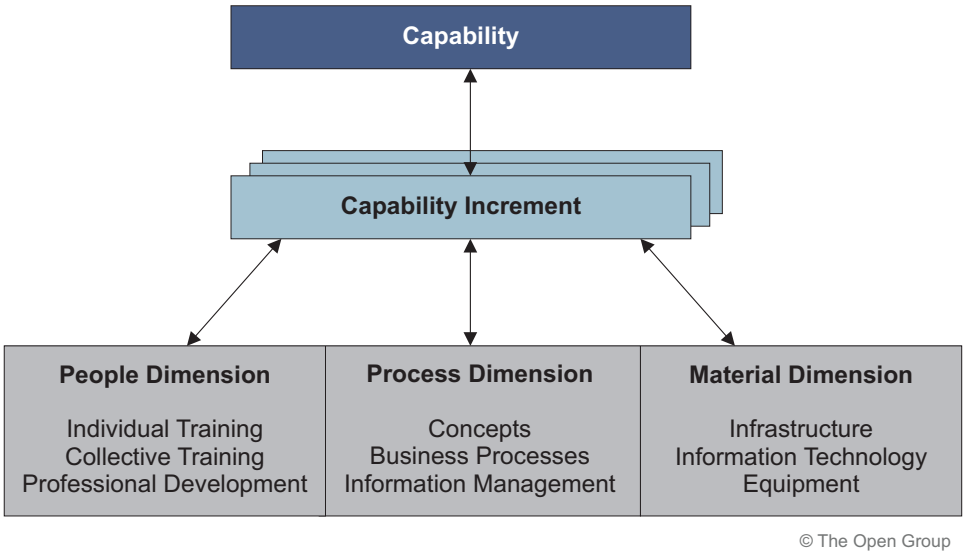

Figure 28-2 Capability Increments and Dimensions

28.3.2 Capability Increments

A capability will take an extended time to deliver (specifics will be a function of the organization and industry vertical) and will normally involve many projects delivering numerous increments. In addition, the capability needs to provide real business value to stakeholders as soon as possible and maintain momentum to achieve the Target Architecture as well as the associated executive support and corporate funding. Therefore, it is useful to break the capability into capability increments that deliver discrete, visible, and quantifiable outcomes as well as providing the focus for Transition Architectures and the deliverables from numerous inter-dependent projects. These outcomes are the Critical Success Factors (CSFs) for continued capability support.

Communicating the potentially complex incremental evolution of a capability to the stakeholder community is essential to establish buy-in at the start and to maintain their buy-in during the transition. The Capability Increment "Radar" diagram (see Figure 28-3) is a proven approach to describing how a capability will evolve over time. The architect selects the aspects of capability that are important to the stakeholder community as lines radiating from the center. Against each line, the architect draws points that represent significant "capability points" ("lower" capability points nearest the center; "higher" capability points farthest from the center). With these "markers" in place the architect can, by joining up the capability points into a closed loop, demonstrate in a simple form how each "capability increment" will extend on the previous increment. This, of course, requires that each capability point is formally defined and "labeled" in a way that is meaningful to the stakeholders. In Figure 28-3, we have depicted Capability Increment 0 as the starting capability.

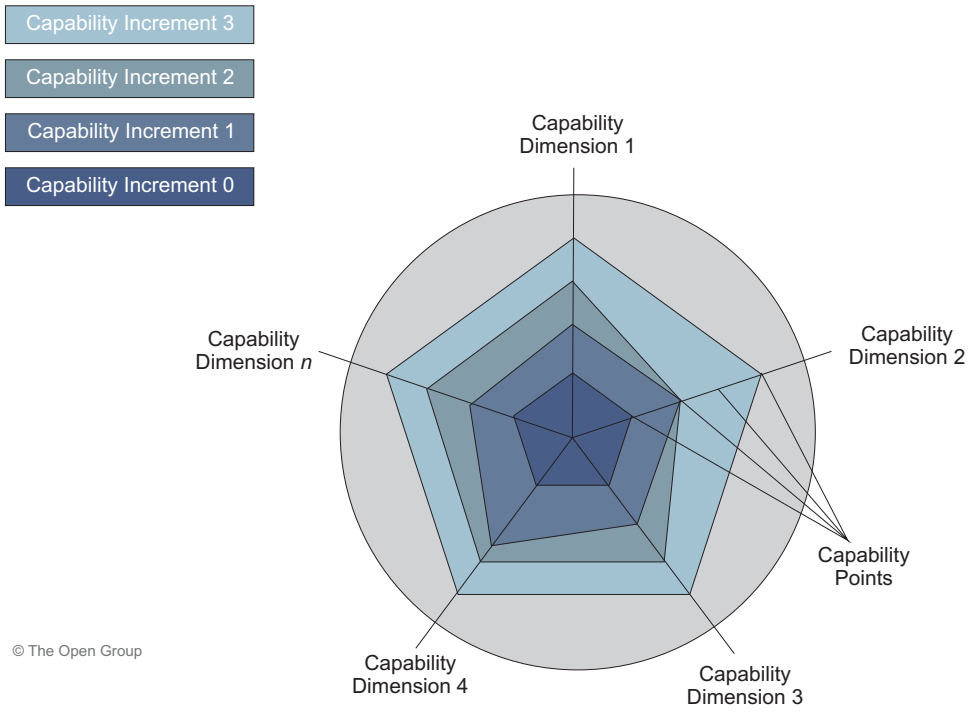

Figure 28-3 Capability Increment Radar

28.4 Capabilities in an Enterprise Architecture Context

The capabilities are directly derived from the corporate strategic plan by the corporate strategic planners that are and/or include the Enterprise Architects and satisfy the enterprise goals, objectives, and strategies. Most organizations will also have an annual business plan that describes how the organization intends to proceed over the next fiscal period in order to meet the enterprise strategic goals.

Figure 28-4 illustrates the crucial relationships between capability-based planning, Enterprise Architecture, and project/portfolio management. On the left-hand side, capability management is aligned with Enterprise Architecture. The key is that all of the architectures will be expressed in terms of business outcomes and value rather than in IT terms (e.g., establishment of a server farm), thereby ensuring IT alignment with the business.

The intent is that the corporate strategic direction drives the Architecture Vision in Phase A, as well as the corporate organization which will be the basis for the creation of portfolios.

Specific capabilities targeted for completion will be the focus of the Architecture Definition (Phases B, C, and D) and, based upon the identified work packages, Phase E projects will be conceived.

The capability increments will be the drivers for the Transition Architectures (Phase E) that will structure the project increments. The actual delivery will be co-ordinated through the Implementation and Migration Plans (Phase F).

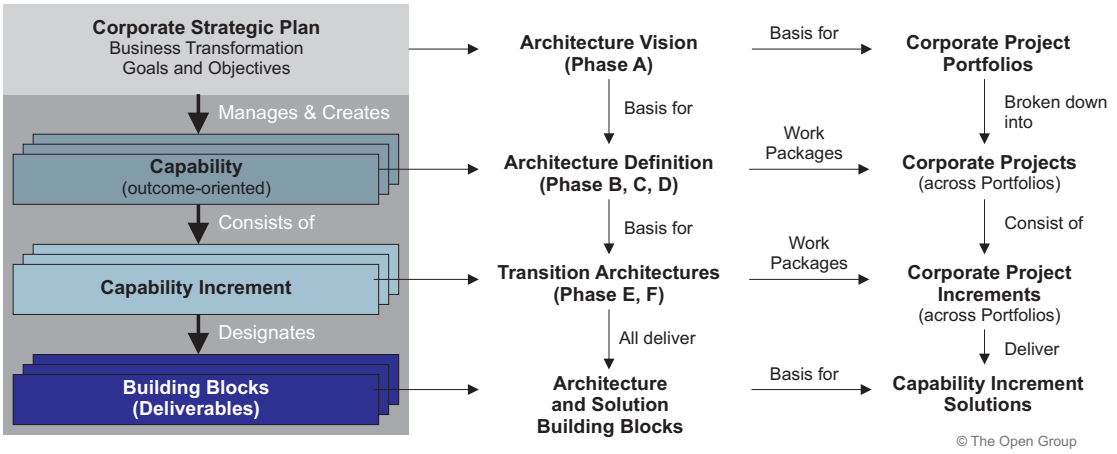

Figure 28-4 Relationship Between Capabilities, Enterprise Architecture, and Projects

Capability managers will perform similar tasks to that of the portfolio managers, but across the portfolios aligning the projects and project increments to deliver continuous business value. Whereas the portfolio managers will be concerned with the co-ordination of their projects to optimally design, build, and deliver the Solution Building Blocks (SBBs). Ideally, capability managers will also manage funding that can use the Transition Architectures as gates. Co-ordination between the portfolio and capability managers will have to be provided at the corporate level.

28.5 Summary

Capability-based planning is a versatile business planning paradigm that is very useful from an Enterprise Architecture perspective. It assists in aligning IT with the business and helps focus IT architects on the continuous creation of business value.

The TOGAF Standard, Version 9.2

Part IV:

Architecture Content Framework

The Open Group

Chapter 29

Introduction to Part IV

29.1 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, project compliance assessments, etc. 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, agreed, 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, business interaction matrix, and a use-case 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 required capability and shape the specification of Solution Building Blocks (SBBs); for example, a customer services capability may be required within an enterprise, supported by many SBBs, such as processes, data, and application software

— **Solution Building Blocks (SBBs)** represent components that will be used to implement the required capability; for example, a network is a building block that can be described through complementary artifacts and then put to use to realize solutions for the enterprise

The relationships between deliverables, artifacts, and building blocks are shown in Figure 29-1.

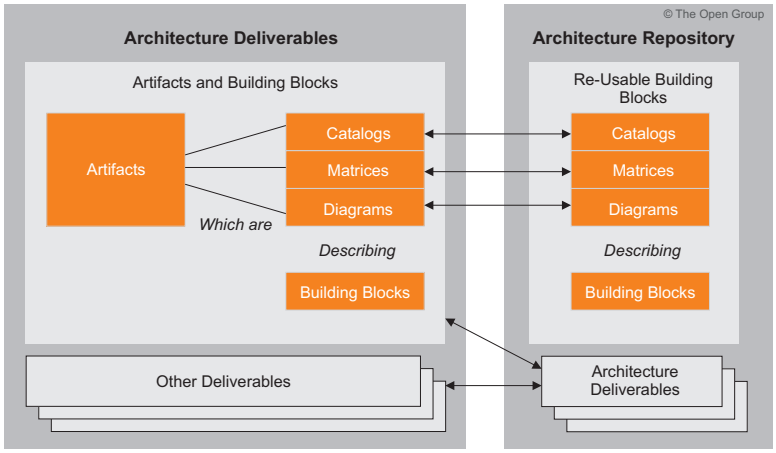

Figure 29-1 Relationships between Deliverables, Artifacts, and Building Blocks

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 29-2.

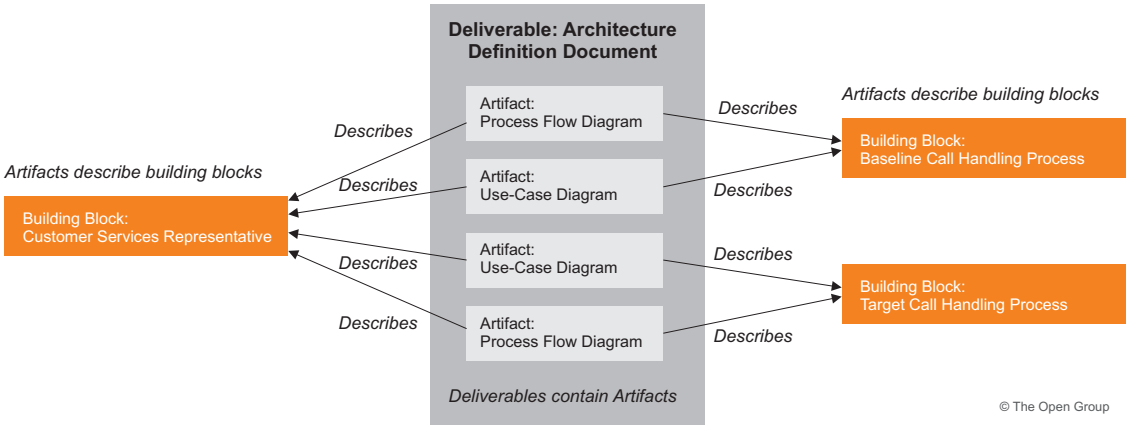

Figure 29-2 Example — Architecture Definition Document

29.2 Content Metamodel

The content metamodel provides a definition of all the types of building blocks that may exist within an architecture, showing how these building blocks can be described and related to one another. For example, when creating an architecture, an architect will identify applications, "data entities" held within applications, and technologies that implement those applications. These applications will in turn support particular groups of business user or actor, and will be used to fulfil "business services".

The content metamodel identifies all of these concerns (i.e., application, data entity, technology, actor, and business service), shows the relationships that are possible between them (e.g., actors consume business services), and finally identifies artifacts that can be used to represent them.

Figure 29-3 shows an overview of the content metamodel.

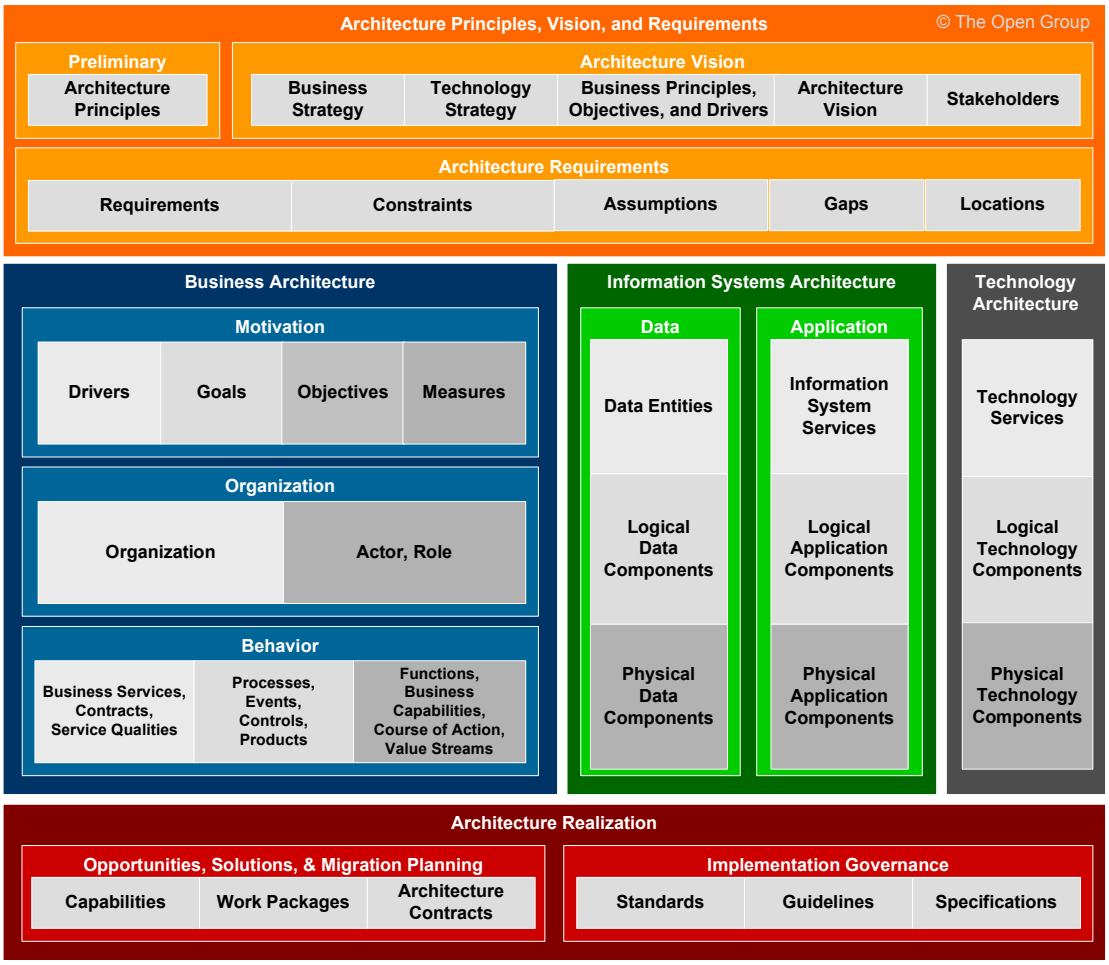

Figure 29-3 Content Metamodel Overview

29.3 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.

29.4 Structure of Part IV

Part IV: Architecture Content Framework is structured as follows:

- Introduction (this chapter)
- Content Metamodel (see Chapter 30)
- Architectural Artifacts (see Chapter 31)
- Architecture Deliverables (see Chapter 32)
- Building Blocks (see Chapter 33)

Chapter 30

Content Metamodel

30.1 Overview

The TOGAF Architecture Development Method (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 metamodel provided here defines 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.

30.2 Content Metamodel Vision and Concepts

This section provides an overview of the objectives of the content metamodel, the concepts that support the metamodel, and an overview of the metamodel itself. Subsequent sections then go on to discuss each area of the metamodel in more detail. Contents of this section are as follows:

- Core content metamodel concepts (see Section 30.2.1) identifies the key concepts within the core content metamodel, including:
  - Core and extension content
  - Formal and informal modeling
  - Core metamodel entities
- Overview of the TOGAF content metamodel (see Section 30.2.2) provides a high-level overview of the content of the metamodel

30.2.1 Core Content Metamodel Concepts

A TOGAF architecture is based on defining a number of architectural building blocks within architecture catalogs, specifying the relationships between those building blocks in architecture matrices, and then presenting communication diagrams that show in a precise and concise way what the architecture is.

This section introduces the core concepts that make up the TOGAF content metamodel, through the following subsections:

- **Core and Extension Content** provides an introduction to the way in which the TOGAF framework employs a basic core metamodel and then applies a number of extension modules to address specific architectural issues in more detail

- **Core Metamodel Entities** introduces the core TOGAF metamodel entities, showing the purpose of each entity and the key relationships that support architectural traceability

Core and Extension Content

The role of the TOGAF framework is to provide an open standard for architecture that is applicable in many scenarios and situations. In order to meet this vision, it is necessary to provide a fully featured Enterprise Architecture metamodel for content and also to provide the ability to avoid carrying out unnecessary activities by supporting tailoring.

The metamodel must provide a basic model with the minimum feature set and then support the inclusion of optional extensions during engagement tailoring.

The core TOGAF content metamodel and its extensions are illustrated in Figure 30-1.

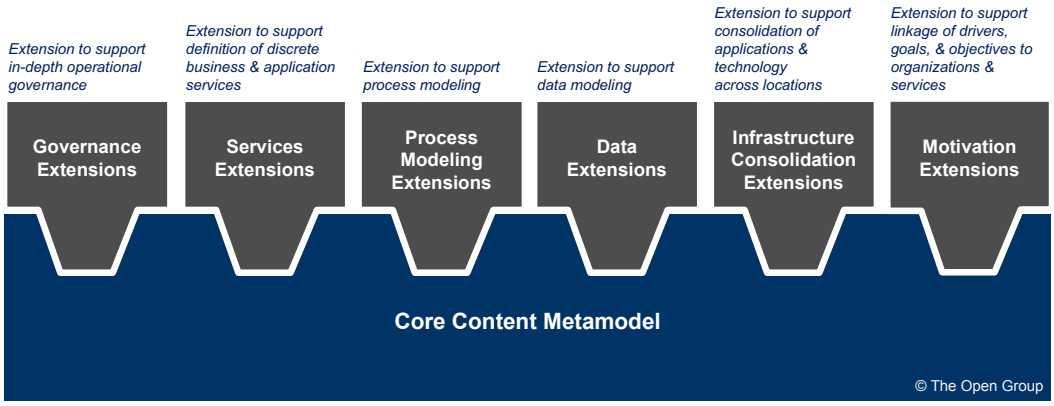

Figure 30-1 TOGAF Content Metamodel and its Extensions

The core metamodel provides a minimum set of architectural content to support traceability across artifacts. Additional metamodel concepts to support more specific or more in-depth modeling are contained within a group of extensions that logically cluster extension catalogs, matrices, and diagrams, allowing focus in areas of specific interest and focus.

All extension modules are optional and should be selected during the Preliminary Phase of the architecture development to meet the needs of the organization. Additionally, the extension groupings described by the content metamodel are only a suggestion and further tailoring may be carried out to suit the specific needs at the discretion of the architects.

This core and extension concept is intended as a move towards supporting formal method extension approaches within the TOGAF framework, such as the method plug-in concept found within the Software Process Engineering Metamodel (SPEM™) developed by the Object Management Group (OMG).[6]

6. Refer to www.omg.org/spec/SPEM.

Core Metamodel Entities

The content metamodel uses the terminology discussed within the TOGAF ADM as the basis for a formal metamodel. The following core terms are used:

- **Actor**: a person, organization, or system that is outside the consideration of the architecture model, but interacts with it
- **Application Component**: an encapsulation of application functionality that is aligned to implementation structure
- **Business Capability**: a particular ability that a business may possess or exchange to achieve a specific purpose
- **Business Service**: supports business capabilities through an explicitly defined interface and is explicitly governed by an organization
- **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**: an encapsulation of data that is recognized by a business domain expert as a discrete concept

  Data entities can be tied to applications, repositories, and services and may be structured according to implementation considerations.
- **Function**: delivers business capabilities closely aligned to an organization, but not explicitly governed by the organization
- **Information System Service**: the automated elements of a business service

  An information system service may deliver or support part or all of one or more business services.
- **Organization Unit**: a self-contained unit of resources with goals, objectives, and measures

  Organization units may include external parties and business partner organizations.
- **Role**: an actor assumes a role to perform a task
- **Technology Component**: an encapsulation of technology infrastructure that represents a class of technology product or specific technology product
- **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 value-adding activities that create an overall result for a customer, stakeholder, or end-user

A more in-depth definition of terms used within the content metamodel can be found in Part I, Chapter 3.

Some of the key relationship concepts related to the core metamodel entities are described below:

- **Process should normally be used to describe flow**

  A process is a flow of interactions between functions and services and cannot be physically deployed. All processes should describe the flow of execution for a function and therefore the deployment of a process is through the function it supports; i.e., an application implements a function that has a process, not an application implements a process.

- **Function describes units of business capability at all levels of granularity**

  The term "function" is used to describe a unit of business capability at all levels of granularity, encapsulating terms such as value chain, process area, capability, business function, etc. Any bounded unit of business function should be described as a function.

- **Business services support organizational objectives and are defined at a level of granularity consistent with the level of governance needed**

  A business service operates as a boundary for one or more functions. The granularity of business services is dependent on the focus and emphasis of the business (as reflected by its drivers, goals, and objectives). A service in Service-Oriented Architecture (SOA) terminology (i.e., a deployable unit of application functionality) is actually much closer to an application service, application component, or technology component, which may implement or support a business service.

- **Business services are deployed onto application components**

  Business services may be realized by business activity that does not relate to IT, or may be realized through IT. Business services that are realized through IT are implemented onto application components. Application components can be hierarchically decomposed and may support one or more business services. It is possible for a business service to be supported by multiple application components, but this is problematic from a governance standpoint and is symptomatic of business services that are too coarse-grained, or application components that are too fine-grained.

- **Application components are deployed onto technology components**

  An application component is implemented by a suite of technology components. For example, an application such as "HR System" would typically be implemented on several technology components, including hardware, application server software, and application services.

Figure 30-2 illustrates the core entities and their relationships.

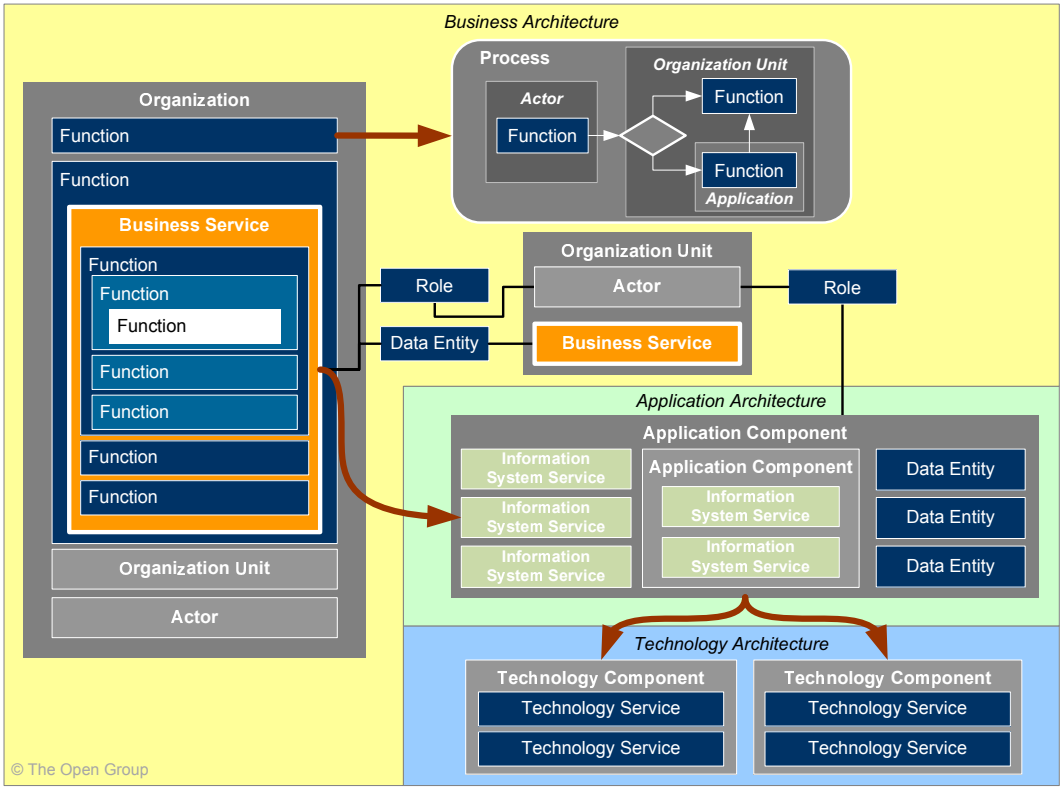

Figure 30-2 Core Entities and their Relationships

30.2.2 Overview of the Content Metamodel

The content metamodel defines a set of entities that allow architectural concepts to be captured, stored, filtered, queried, and represented in a way that supports consistency, completeness, and traceability.

At the highest level, the content framework is divided up in line with the TOGAF ADM phases, as shown in Figure 30-3.

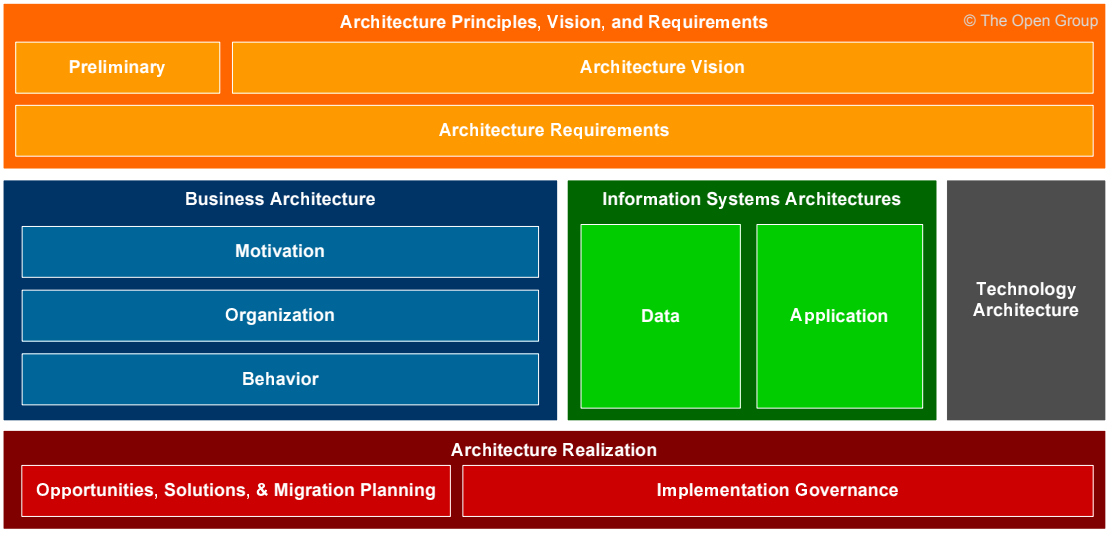

Figure 30-3 Content Framework by ADM Phases

- **Architecture Principles, Vision, and Requirements** entities 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 architecture context is typically collected in the Preliminary and Architecture Vision phases.

- **Business Architecture** entities capture architectural models of business operation, looking specifically at factors that motivate the enterprise, how the enterprise is organizationally structured, and also what business capabilities the enterprise has

- **Information Systems Architecture** entities capture architecture models of IT systems, looking at applications and data in line with the TOGAF ADM phases

- **Technology Architecture** entities capture procured technology assets that are used to implement and realize information system solutions

- **Architecture Realization** entities capture change roadmaps showing transition between architecture states and binding statements that are used to steer and govern an implementation of the architecture

A more detailed representation of the content metamodel is shown in Figure 30-4.

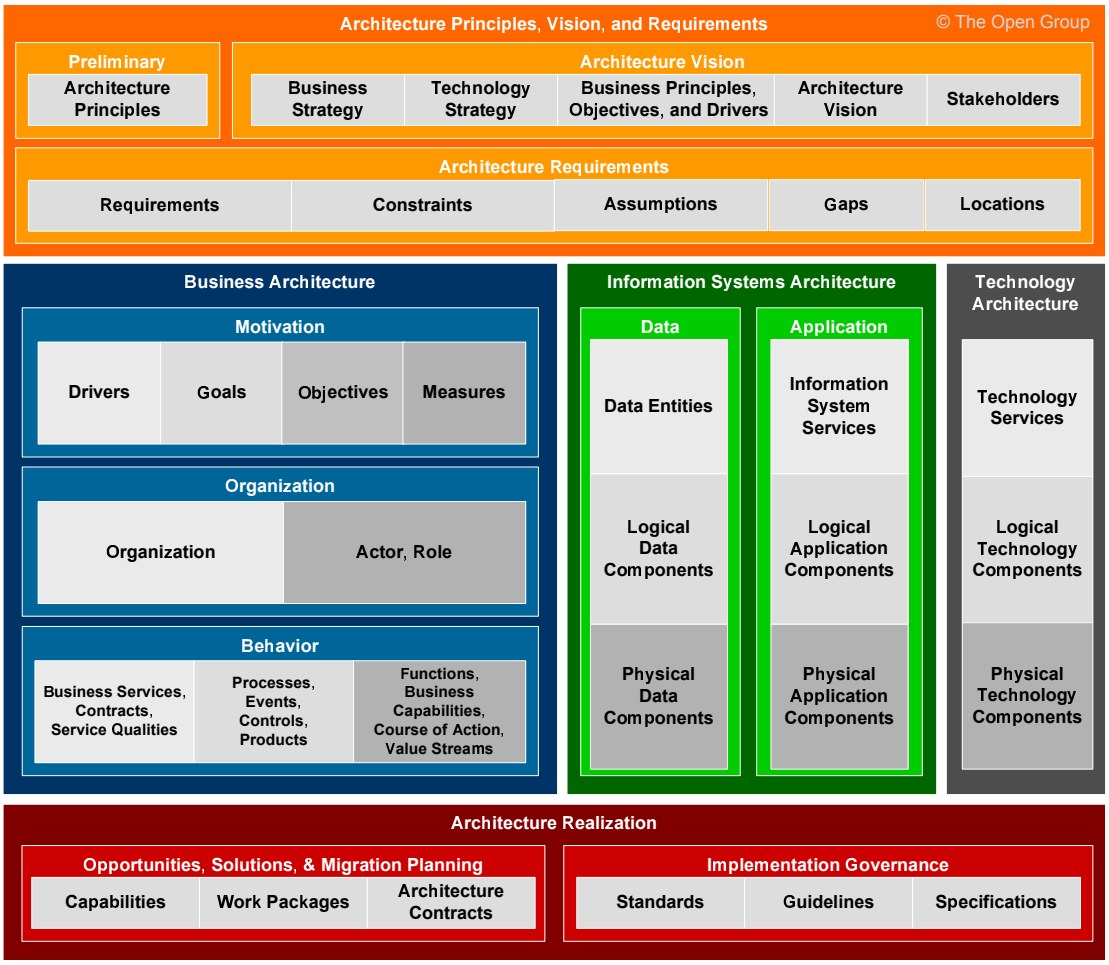

Figure 30-4 Detailed Representation of the Content Metamodel

30.3 Content Metamodel in Detail

This section contains the following subsections:

- Core Content Metamodel (see Section 30.3.1) describes the metamodel entities that form the core content metamodel
- Full Content Metamodel (see Section 30.3.2) describes the metamodel entities that form extensions to the content metamodel

30.3.1 Core Content Metamodel

Figure 30-5 shows the metamodel entities and relationships that are present within the core content metamodel.

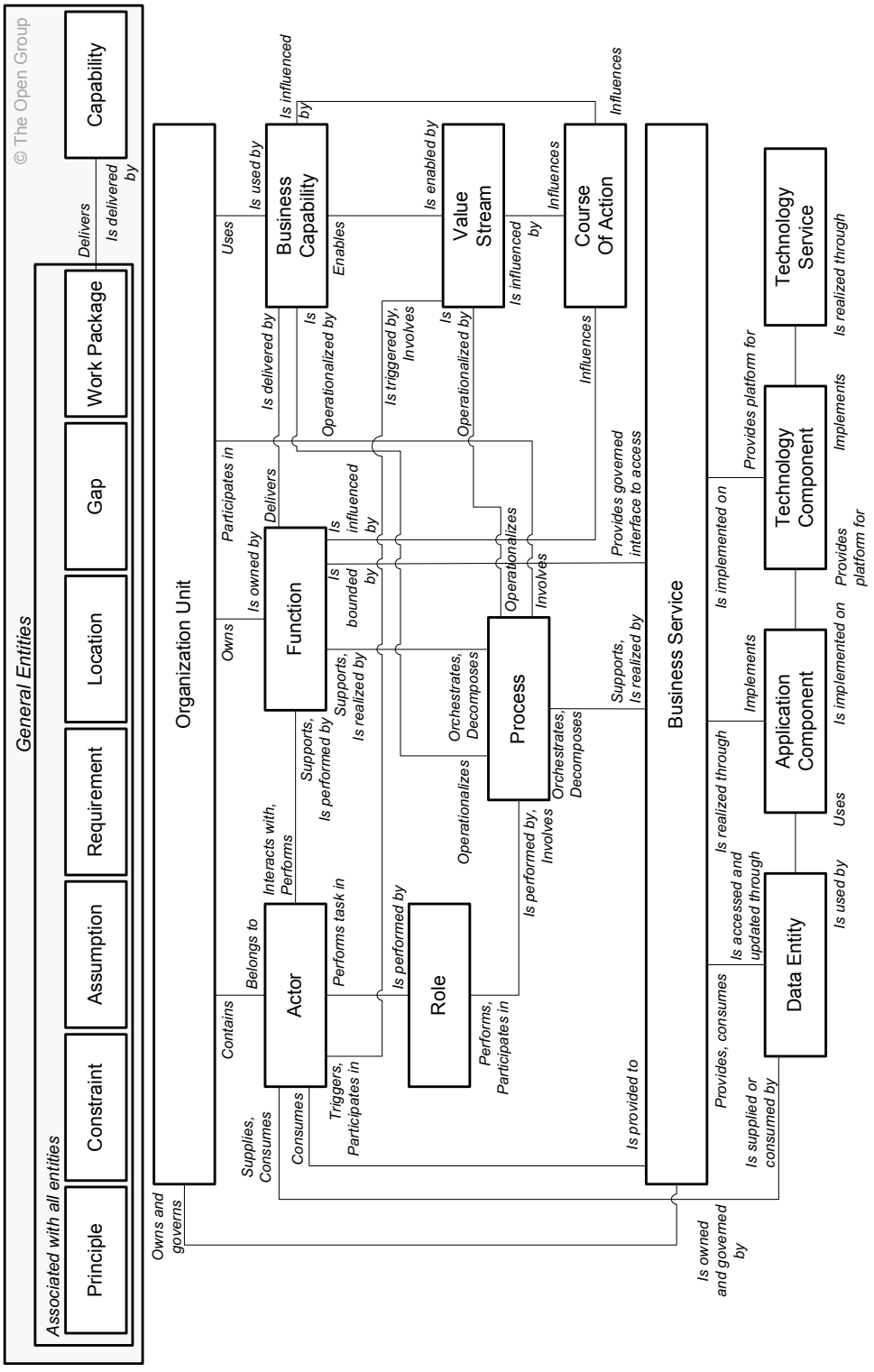

Figure 30-5 Entities and Relationships Present within the Core Content Metamodel

30.3.2 Full Content Metamodel

When all extensions are applied to the core content metamodel, a number of additional metamodel entities are introduced. Figure 30-6 shows which entities are contained in the core content metamodel and which entities are introduced by which extension.

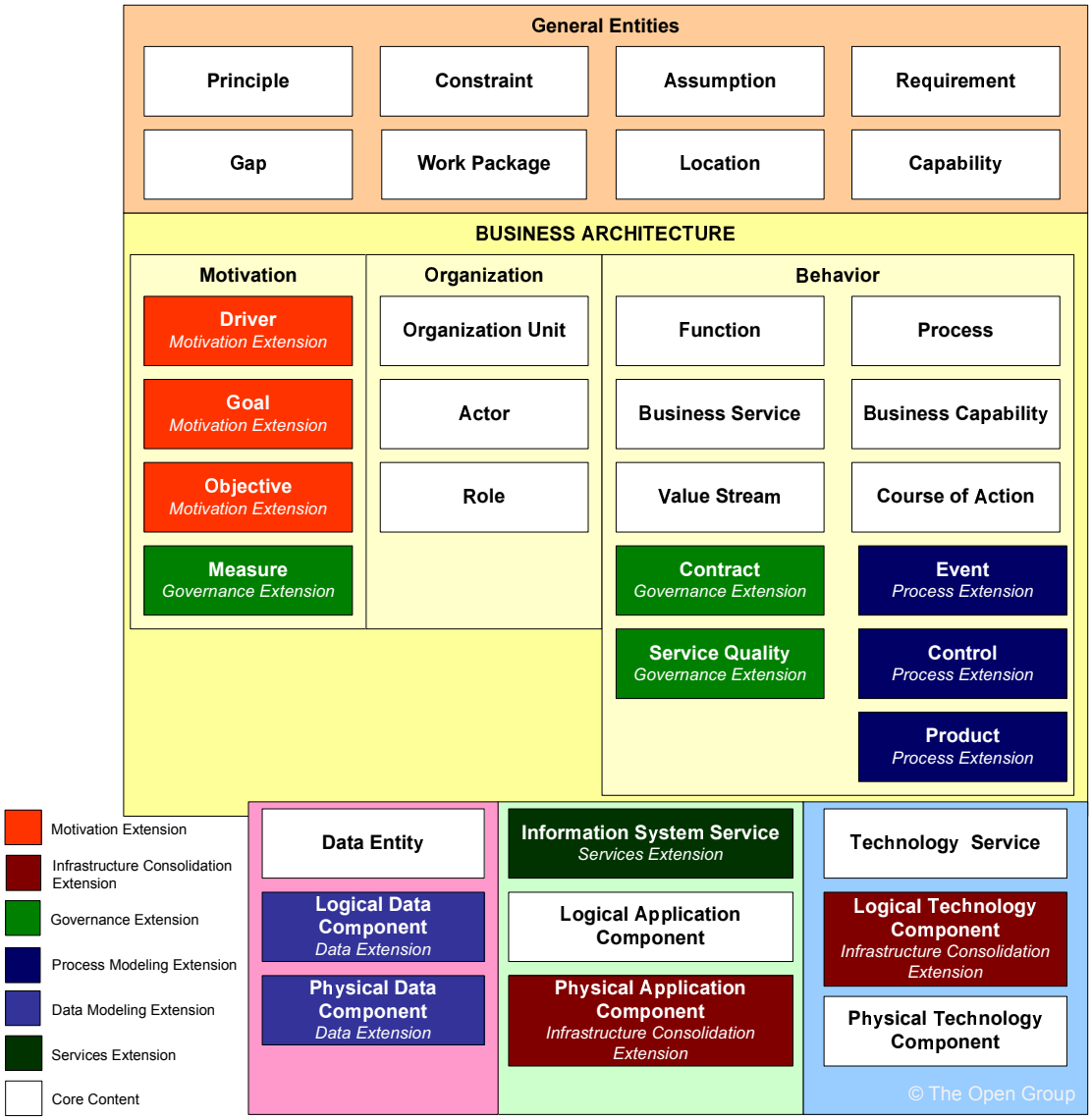

Figure 30-6 Content Metamodel with Extensions

The relationships between entities in the full metamodel are shown in Figure 30-7.

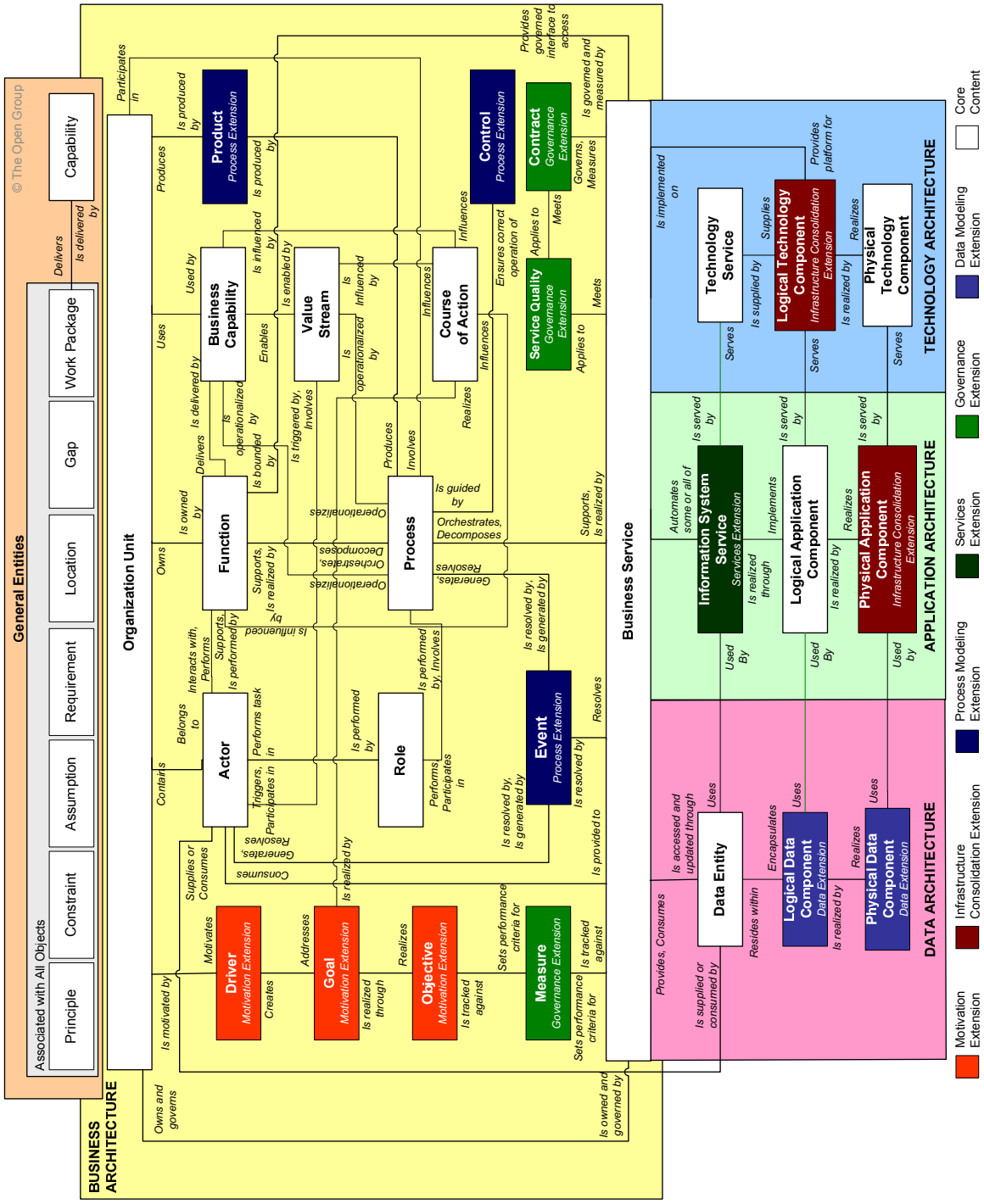

Figure 30-7 Relationships between Entities in the Full Metamodel

30.4 Content Metamodel Extensions

As discussed earlier, the TOGAF content metamodel supports a number of extension modules that allow more in-depth consideration for particular architecture concerns. Figure 30-8 shows the core content metamodel and predefined extension modules.

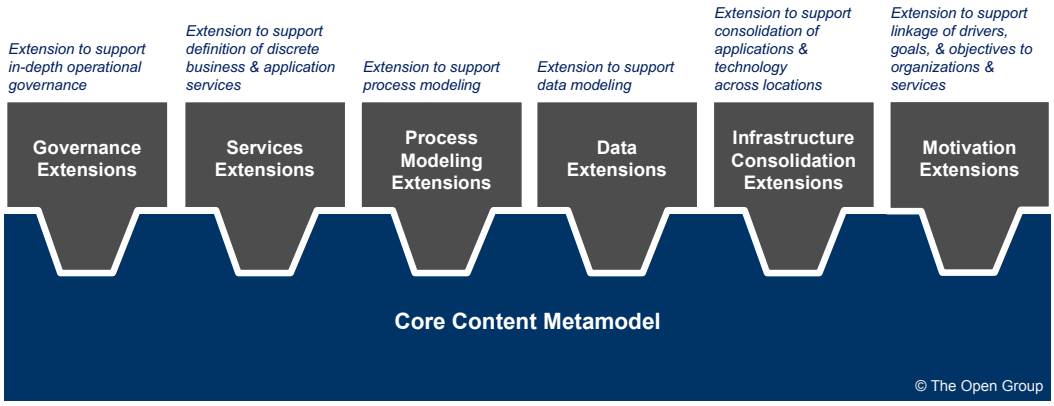

Figure 30-8 Core Content Metamodel and Predefined Extension Modules

During the Architecture Vision phase of a particular engagement, the scope of the engagement will be used to make a determination on appropriate extensions to be employed in order to adequately address the architecture requirements. For example, the scope of an engagement could be defined as core content, plus the governance extensions, as shown in Figure 30-9.

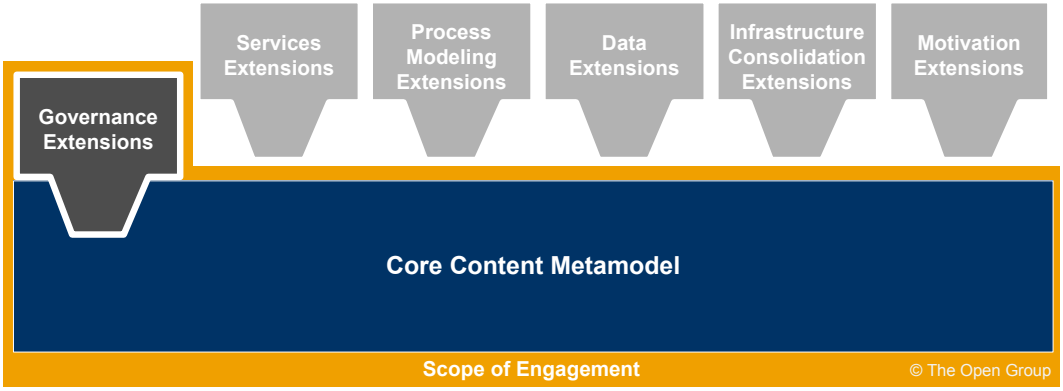

Figure 30-9 Core Content with Governance Extensions

The following sections provide a more detailed description of the purpose and content of each of the extension modules.

30.4.1 Governance Extensions

Purpose

The governance extension is intended to allow additional structured data to be held against objectives and business services, supporting operational governance of the landscape.

The scope of this extension is as follows:

- The ability to apply measures to objectives and then link those measures to services
- The ability to apply contracts to service communication or service interactions with external users and systems
- The ability to define re-usable service qualities defining a service-level profile that can be used in contracts
- Creation of additional diagrams to show ownership and management of systems

This extension should be used in the following situations:

- When an organization is considering IT change that will result in a significant impact to existing operational governance models
- When an organization has granular requirements for service levels that differ from service to service
- When an organization is looking to transform its operational governance practice
- When an organization has very strong focus on business drivers, goals, and objectives and how these trace to service levels

The benefits of using this extension are as follows:

- Service levels are defined in a more structured way, with:
  - More detail
  - The ability to re-use service profiles across contracts
  - Stronger tracing to business objectives
- Impacts to operations and operational governance models are considered in a more structured way, with:
  - Additional diagrams of system and data ownership
  - Additional diagrams of system operation and dependencies on operations processes

In addition to the extensions described here, organizations wishing to focus on Architecture Governance should also consult:

- The COBIT framework for IT governance provided by the Information Systems Audit and Control Association (ISACA); refer to www.isaca.org
- The IT Portfolio Management Facility™ (ITPMF™) from the OMG; refer to www.omg.org/spec/ITPMF

Required Extensions to the Core Metamodel

Extensions to the core metamodel entities and relationships are shown in Figure 30-10.

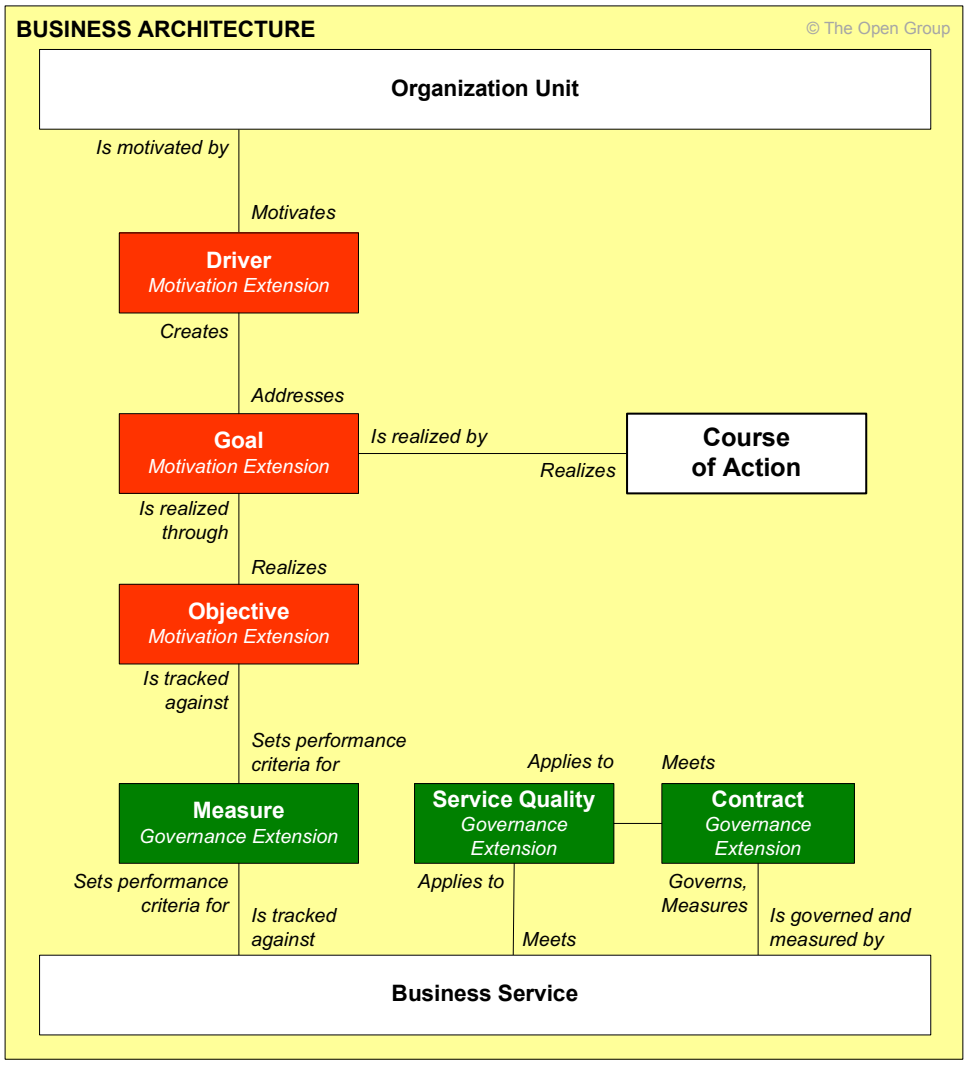

Figure 30-10 Governance Extensions: Extensions to Core Metamodel

Changes to the metamodel entities and relationships are as follows:

- Measure is added as an entity that links objective and business service
- Service Quality is added as an entity that provides a generic service profile template to be applied to business services or contracts
- Contract is added as an entity that formalizes the functional and non-functional characteristics of a service interaction with other services, external applications, or users

Changes to the metamodel attributes are as follows:

- Attributes are added for the metamodel entities of Measure, Service Quality, and Service Contract

Additional diagrams to be created are as follows:

- Enterprise Manageability diagram

30.4.2 Services Extensions

Purpose

The services extension is intended to allow more sophisticated modeling of the service portfolio by creating a concept of Information System (IS) services in addition to the core concept of business services. IS services are directly supported by applications and creating the layer of abstraction relaxes the constraints on business services while simultaneously allowing technical stakeholders to put more formality into an IS service catalog.

The scope of this extension is as follows:

- Creation of IS services as an extension of business service

This extension should be used in the following situations:

- When the business has a preset definition of its services that does not align well to technical and architectural needs
- When business and IT use different language to describe similar capabilities
- Where IT service is misaligned with business need, particularly around the areas of quality of service, visibility of performance, and management granularity
- Where IT is taking initial steps to engage business in discussions about IT architecture

The benefits of using this extension are as follows:

- Business services can be defined outside of the constraints that exist in the core metamodel; this allows for a more natural engagement with business stakeholders
- IS services can be defined according to a model that maps closely to implementation, providing a more realistic solution abstraction to support IT decision-making
- Business and IS service relationships show where the business view aligns with the IS view and where there are misalignments

In addition to the extensions described here, organizations wishing to focus on services-centric architectures should also consult:

- The Service Component Architecture (SCA) specification developed by the Open Service Oriented Architecture (OSOA) collaboration; refer to www.oasis-opencsa.org/sca
- The Service Data Objects (SDO) specification developed by the Open Service Oriented Architecture (OSOA) collaboration; refer to www.oasis-opencsa.org/sdo

Required Extensions to the Core Metamodel

Extensions to the core metamodel entities and relationships are shown in Figure 30-11.

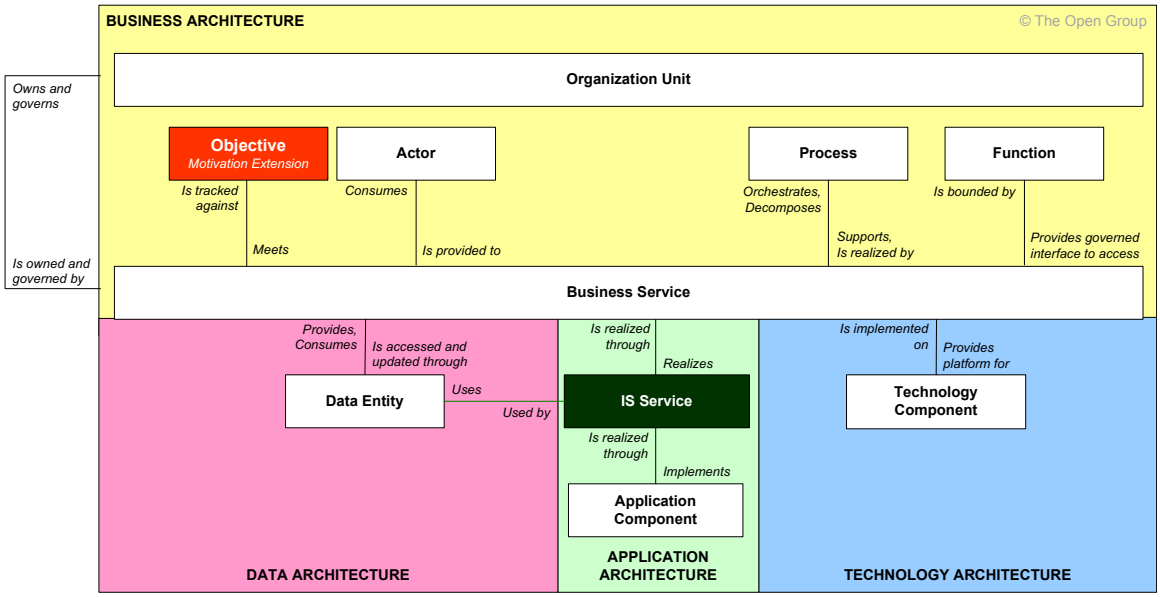

Figure 30-11 Services Extension: Extensions to Core Metamodel

Changes to the metamodel entities and relationships are as follows:

- IS Service is added as a metamodel entity, extending business service
- IS Service inherits all the relationships of a business service
- A relationship is created linking an IS service to a business service

Changes to the metamodel attributes are as follows:

- IS Service is added as a type of business service

Additional diagrams to be created are as follows:

- Business Use-Case Diagram
- Organization Decomposition Diagram

30.4.3 Process Modeling Extensions

Purpose

The process modeling extension is intended to allow detailed modeling of process flows by adding events, products, and controls to the metamodel. Typically, Enterprise Architecture does not drill into process flow, but in certain process-centric or event-centric organizations it may be necessary to elaborate process in a much more formal manner using this extension module.

The scope of this extension is as follows:

- Creation of events as triggers for processes
- Creation of controls that business logic and governance gates for process execution
- Creation of products to represent the output of a process
- Creation of event diagrams to track triggers and state changes across the organization

This extension should be used in the following situations:

- Where the architecture must pay specific attention to state and events
- Where the architecture is required to explicitly identify and store process control steps; for example, to support regulatory compliance
- Where the architecture features critical or elaborate process flows

The benefits of using this extension are as follows:

- This extension allows detailed process modeling and the cataloging of process artifacts
- May be used to support regulatory compliance activities
- May be used to re-purpose legacy or non-architectural process decomposition analysis

In addition to the extensions described here, organizations wishing to focus on process-centric architectures should also consult:

- The Business Process Modeling Notation (BPMN) specification, provided by the OMG; refer to www.bpmn.org
- The Software Process Engineering Metamodel (SPEM) specification, provided by the OMG; refer to www.omg.org/spec/SPEM

Required Extensions to the Core Metamodel

Extensions to the core metamodel entities and relationships are shown in Figure 30-12.

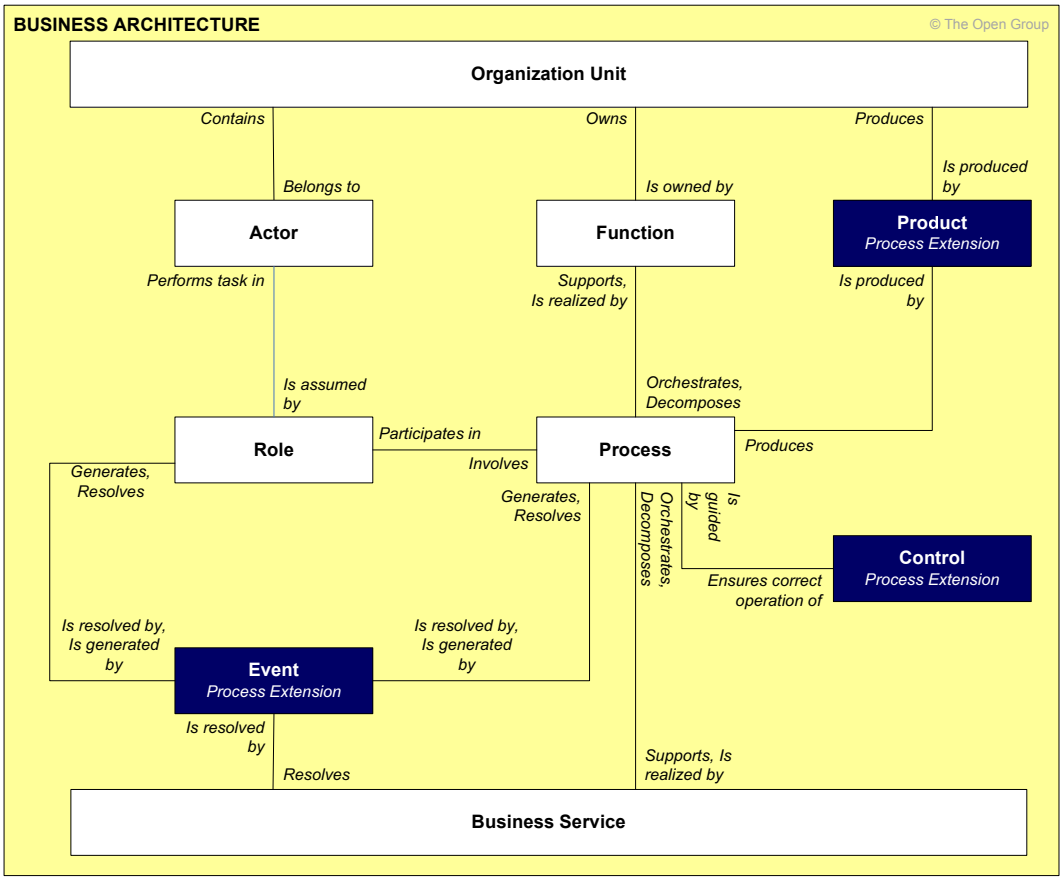

Figure 30-12 Process Modeling Extensions: Extensions to Core Metamodel

Changes to the metamodel entities and relationships are as follows:

- Event is added as a metamodel entity, sitting between Actor, Process, and Service
- Control is added as a metamodel entity, relating to a Process
- Product is added as a metamodel entity, linking Organization and Processes

Changes to the metamodel attributes are as follows:

- Attributes are added for the metamodel entities of Event, Control, and Product

Additional diagrams to be created are as follows:

- Process Flow diagrams, showing the way in which business functions, events, controls, and products are linked to support a particular business scenario
- Event diagrams, showing events, were they are received from, and what processes they trigger

30.4.4 Data Extensions

Purpose

The data extension is intended to allow more sophisticated modeling and the encapsulation of data. The core model provides a data entity concept which supports the creation of data models, which is then extended by this extension to include the concept of a data component. Data components form a logical or physical encapsulation of abstract data entities into units that can be governed and deployed into applications.

The scope of this extension is as follows:

- Creation of logical data components that group data entities into encapsulated modules for governance, security, and deployment purposes
- Creation of physical data components that implement logical data components and are analogous to databases, registries, repositories, schemas, and other techniques of segmenting data
- Creation of data lifecycle, data security, and data migration diagrams of the architecture to show data concerns in more detail

This extension should be used in the following situations:

- Where the architecture features significant complexity and risk around the location, encapsulation, and management of or access to data

The benefits of using this extension are as follows:

- The structure of data is modeled independently from its location, allowing data models to be developed that span multiple systems without being tied to physical concerns
- Logical groupings of data can be used to set governance, security, or deployment boundaries around data, providing a much more holistic appreciation of data issues surrounding the architecture

Required Extensions to the Core Metamodel

Extensions to the core metamodel entities and relationships are shown in Figure 30-13.

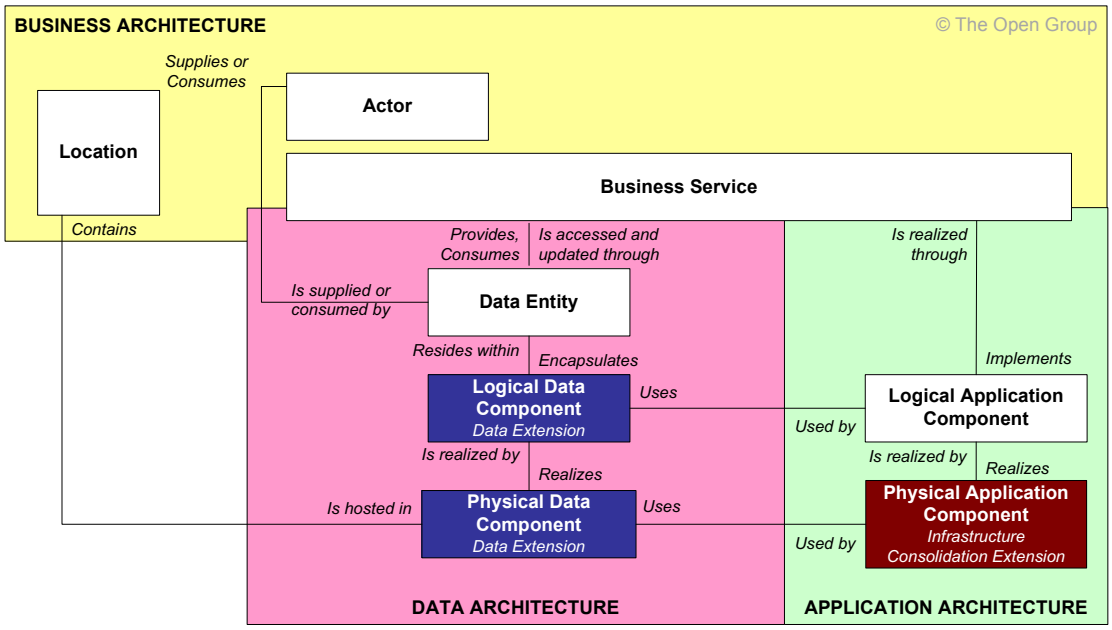

Figure 30-13 Data Extensions: Extensions to Core Metamodel

Changes to the metamodel entities and relationships are as follows:

- Logical Data Component is added as a metamodel entity, encapsulating data entities
- Physical Data Component is added as a metamodel entity, extending Logical Data Component
- A relationship is created between Physical Data Component and Application Component; if the infrastructure consolidation extension is applied, this should be to Physical Application Component

Changes to the metamodel attributes are as follows:

- Attributes are added for the metamodel entities of Logical Data Component and Physical Data Component

Additional diagrams to be created are as follows:

- Data Security diagram
- Data Migration diagram
- Data Lifecycle diagram

30.4.5 Infrastructure Consolidation Extensions

Purpose

The infrastructure consolidation extension is intended to be used in landscapes where the application and technology portfolios have become fragmented and the architecture seeks to consolidate the business as usual capability into a smaller number of locations, applications, or technology components.

The scope of this extension is as follows:

- Creation of logical and physical application components to abstract the capability of an application away from the actual applications in existence
- Creation of logical and physical technology components to abstract product type from the actual technology products in existence
- Creation of additional diagrams focusing on the location of assets, compliance with standards, structure of applications, application migration, and infrastructure configuration

This extension should be used in the following situations:

- Where many technology products are in place with duplicate or overlapping capability
- Where many applications are in place with duplicate or overlapping functionality
- Where applications are geographically dispersed and the decision logic for determining the location of an application is not well understood
- When applications are going to be migrated into a consolidated platform
- When application features are going to be migrated into a consolidated application

The benefits of using this extension are as follows:

- Allows visibility and analysis of redundant duplication of capability in the application and technology domains
- Supports analysis of standards compliance
- Supports analysis of migration impact of application or technology consolidation
- Supports detailed architectural definition of application structure

In addition to the extensions described here, organizations wishing to focus on infrastructure consolidation should also consult:

- The Unified Modeling Language (UML), provided by the OMG; refer to www.uml.org
- The Systems Modeling Language™ (SysML®) — www.sysml.org — which reduces the complexity and software engineering focus of UML for the purposes of systems modeling
- The IT Portfolio Management Facility (ITPMF) from the OMG; refer to www.omg.org/spec/ITPMF

Required Extensions to the Core Metamodel

Extensions to the core metamodel entities and relationships are shown in Figure 30-14.

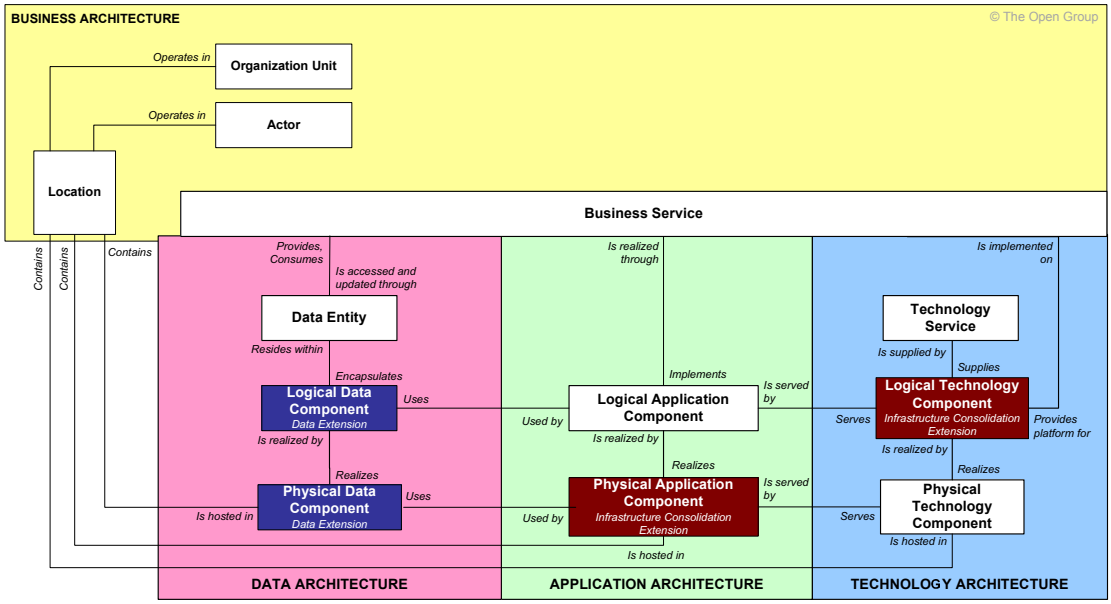

Figure 30-14 Infrastructure Consolidation Extensions: Extensions to Core Metamodel

Changes to the metamodel entities and relationships are as follows:

- Application Components are extended to include Logical Application Components (a class of application) and Physical Application Components (an actual application)
- Technology Components are extended to include Logical Technology Components (a class of technology product) and Physical Technology Components (an actual technology product)

Changes to the metamodel attributes are as follows:

- Creation of attributes for the Metamodel entities of Logical Application Component, Physical Application Component, Logical Technology Component, Physical Technology Component, and Location

Additional diagrams to be created are as follows:

- Process/Application Realization diagram
- Software Engineering diagram
- Application Migration diagram
- Software Distribution diagram
- Processing diagram
- Networked Computing/Hardware diagram
- Network and Communications diagram

30.4.6 Motivation Extensions

Purpose

The motivation extension is intended to allow additional structured modeling of the drivers, goals, and objectives that influence an organization to provide business services to its customers. This in turn allows more effective definition of service contracts and better measurement of business performance.

The scope of this extension is as follows:

- Creation of a metamodel entity for Driver that shows factors generally motivating or constraining an organization
- Creation of a metamodel entity for Goal that shows the strategic purpose and mission of an organization
- Creation of a metamodel entity for Objective that shows near to mid-term achievements that an organization would like to attain
- Creation of a Goal/Objective/Service diagram showing the traceability from drivers, goals, and objectives through to services

This extension should be used in the following situations:

- When the architecture needs to understand the motivation of organizations in more detail than the standard business or engagement principles and objectives that are informally modeled within the core content metamodel
- When organizations have conflicting drivers and objectives and that conflict needs to be understood and addressed in a structured form
- When service levels are unknown or unclear

The benefits of using this extension are as follows:

- Highlights misalignment of priorities across the enterprise and how these intersect with shared services (e.g., some organizations may be attempting to reduce costs, while others are attempting to increase capability)
- Shows competing demands for business services in a more structured fashion, allowing compromise service levels to be defined

In addition to the extensions described here, organizations wishing to focus on architecture modeling of business motivation should also consult:

- The Business Motivation Model™ (BMM™) specification, provided by the OMG; refer to www.omg.org/spec/BMM/About-BMM

Required Extensions to the Core Metamodel

Extensions to the core metamodel entities and relationships are shown in Figure 30-15.

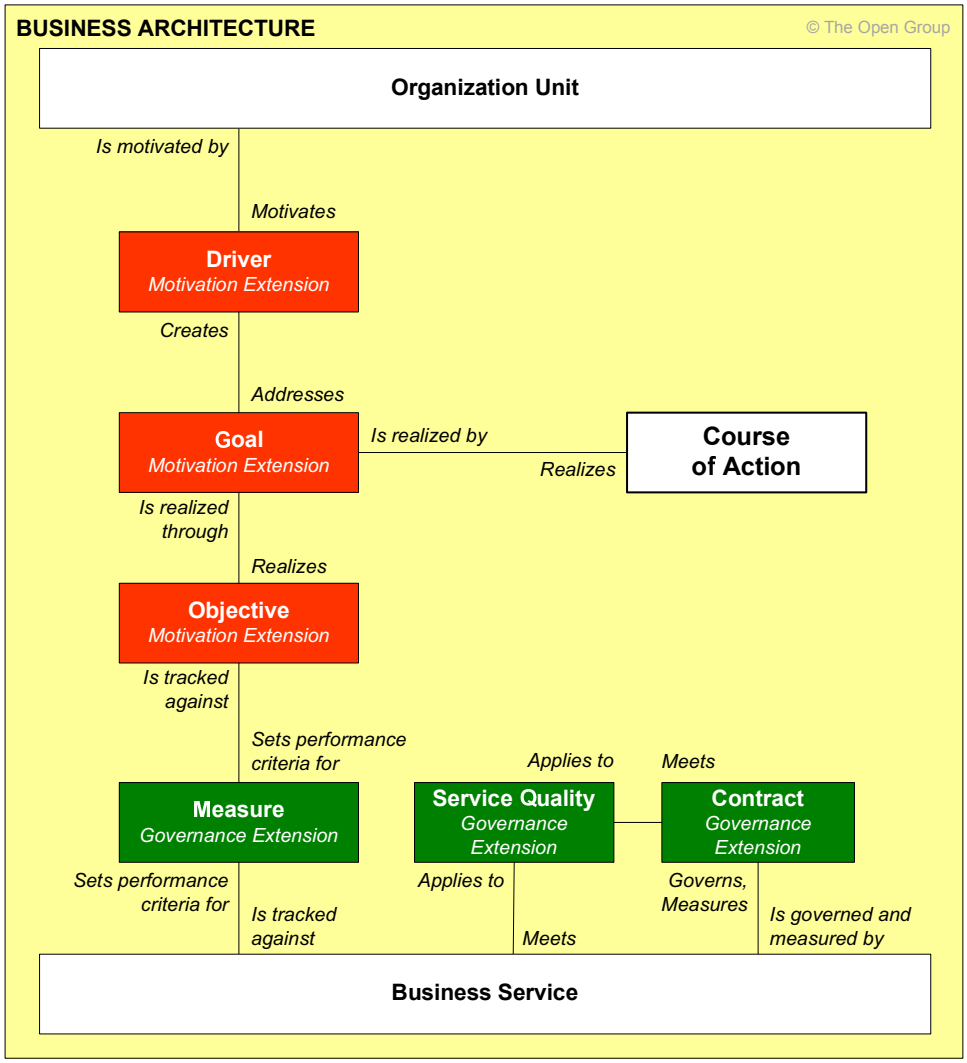

Figure 30-15 Motivation Extensions: Extensions to Core Metamodel

Changes to the metamodel entities and relationships are as follows:

- Driver, Goal, and Objective are added as entities that link Organization Unit to Business Service

Changes to the metamodel attributes are as follows:

- Attributes are added for the metamodel entities of Driver, Goal, and Objective

Additional diagrams to be created are as follows:

- Goal/Objective/Service diagram

30.5 Content Metamodel Entities

The following table lists and describes the entities within the content 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 Component	An encapsulation of application functionality aligned to implementation structure. For example, a purchase request processing application. See also *Logical Application Component* and *Physical Application Component*.
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 Service	Supports business capabilities through an explicitly defined interface and is explicitly governed by an organization.
Capability	A business-focused outcome that is delivered by the completion of one or more work packages. Using a capability-based planning approach, change activities can be sequenced and grouped in order to provide continuous and incremental business value.
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 service consumer and a service provider that establishes functional and non-functional parameters for interaction.
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	An encapsulation of data that is recognized by a business domain expert as a thing. Logical data entities can be tied to applications, repositories, and services and may be structured according to implementation considerations.

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	Delivers business capabilities closely aligned to an organization, but not necessarily explicitly governed by the organization. Also referred to as "business function".
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 Part III, Chapter 23.
Goal	A high-level statement of intent or direction for an organization. Typically used to measure success of an organization.
Information System Service	The automated elements of a business service. An information system service may deliver or support part or all of one or more business services.
Location	A place where business activity takes place and can be hierarchically decomposed.
Logical Application Component	An encapsulation of application functionality that is independent of a particular implementation. For example, the classification of all purchase request processing applications implemented in an enterprise.
Logical Data Component	A boundary zone that encapsulates related data entities to form a logical location to be held; for example, external procurement information.
Logical Technology Component	An encapsulation of technology infrastructure that is independent of a particular product. A class of technology product; for example, supply chain management software as part of an Enterprise Resource Planning (ERP) suite, or a Commercial Off-The-Shelf (COTS) purchase request processing enterprise service.
Measure	An indicator or factor that can be tracked, usually on an ongoing basis, to determine success or alignment with objectives and goals.
Objective	A time-bounded milestone for an organization used to demonstrate progress towards a goal; for example, "Increase capacity utilization by 30% by the end of 2019 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	An application, application module, application service, or other deployable component of functionality. For example, a configured and deployed instance of a Commercial Off-The-Shelf (COTS) Enterprise Resource Planning (ERP) supply chain management application.

Metamodel Entity	Description
Physical Data Component	A boundary zone that encapsulates related data entities to form a physical location to be held. For example, a purchase order business object, comprising purchase order header and item business object nodes.
Physical Technology Component	A specific technology infrastructure product or technology infrastructure product instance. For example, a particular product version of a Commercial Off-The-Shelf (COTS) solution, or a specific brand and version of server.
Principle	A qualitative statement of intent that should be met by the architecture. Has at least a supporting rationale and a measure of importance. **Note:** A sample set of Architecture Principles is defined in Part III, Chapter 20.
Process	A process represents flow of control between or within functions and/or services (depends on the granularity of definition). Processes represent a sequence of activities that together achieve a specified outcome, can be decomposed into sub-processes, and can show operation of a function or service (at next level of detail). Processes may also be used to link or compose organizations, functions, services, and processes.
Product	Output generated by the business. The business product of the execution of a process.
Requirement	A quantitative statement of business need that must be met by a particular architecture or work package.
Role	The usual or expected function of an actor, or the part somebody or something plays in a particular action or event. An actor may have a number of roles. See also *Actor*.
Service	An element of behavior that provides specific functionality in response to requests from actors or other services. A service delivers or supports business capabilities, has an explicitly defined interface, and is explicitly governed. Services are defined for business, information systems, and platforms.
Service Quality	A preset configuration of non-functional attributes that may be assigned to a service or service contract.
Technology Component	An encapsulation of technology infrastructure that represents a class of technology product or specific technology product.
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 value-adding 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.

30.6 Content 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 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	Statement of what tests will be carried out to ensure that the requirement will be met.
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 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.
Contract	Behavior characteristics	Functional behavior to be supported within the scope of the contract.

Metamodel Entity Attribute	Description	
	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.
	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.

Metamodel Entity Attribute	Description	
	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.
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.

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.
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.
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.

Metamodel Entity Attribute	Description	
	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.
Information System 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.
	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.

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.
	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.
	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.

Metamodel Entity Attribute	Description	
	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.
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.

Metamodel Entity Attribute	Description	
	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.
	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 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.

Metamodel Entity Attribute	Description	
	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.
	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	The following categories of work package apply: Work Package, Work Stream, Project, Program, Portfolio.
	Capability delivered	Describes the contribution this work package makes to capability delivery.

30.7 Metamodel Relationships

Source Entity	Target Entity	Name	Extension Module
Actor	Event	Generates	Process
Actor	Event	Resolves	Process
Actor	Function	Interacts with	Core
Actor	Function	Performs	Core
Actor	Value Stream	Participates in	Core
Actor	Value Stream	Triggers	Core
Actor	Organization Unit	Belongs to	Core
Actor	Role	Performs task in	Core
Actor	Service	Consumes	Core
Actor	Actor	Decomposes	Core
Actor	Data Entity	Supplies/Consumes	Core
Business Capability	Course of Action	Is influenced by	Core
Business Capability	Function	Is delivered by	Core
Business Capability	Organizational Unit	Is used by	Core
Business Capability	Process	Is operationalized by	Core
Business Capability	Value Stream	Enables	Core
Capability	Work Package	Is delivered by	Core
Contract	Service	Governs and Measures	Governance
Contract	Service Quality	Meets	Governance
Control	Process	Ensures correct operation of	Process
Course of Action	Business Capability	Influences	Core
Course of Action	Goal	Realizes	Motivation
Course of Action	Function	Influences	Core
Course of Action	Value Stream	Influences	Core
Data Entity	Logical Data Component	Resides within	Data
Data Entity	Service	Is accessed and updated through	Core
Data Entity	Information System Service	Uses	Services
Data Entity	Data Entity	Decomposes	Core
Data Entity	Data Entity	Relates to	Core
Driver	Goal	Creates	Motivation
Driver	Organization Unit	Motivates	Motivation
Driver	Driver	Decomposes	Motivation
Event	Actor	Is resolved by	Process
Event	Actor	Is generated by	Process
Event	Process	Is resolved by	Process
Event	Process	Is generated by	Process
Event	Service	Is resolved by	Process
Function	Actor	Supports	Core
Function	Actor	Is performed by	Core
Function	Business Capability	Delivers	Core
Function	Organization Unit	Is owned by	Core
Function	Process	Supports	Core

Source Entity	Target Entity	Name	Extension Module
Function	Process	Is realized by	Core
Function	Service	Is bounded by	Core
Function	Function	Decomposes	Core
Function	Function	Communicates with	Core
Goal	Course of Action	Is realized by	Motivation
Goal	Driver	Addresses	Motivation
Goal	Objective	Is realized through	Motivation
Goal	Goal	Decomposes	Motivation
Information System Service	Data Entity	Used by	Services
Information System Service	Service	Realizes	Services
Information System Service	Technology Service	Is served by	Services
Information System Service	Logical Application Component	Is realized through	Services
Logical Application Component	Physical Application Component	Is realized by	Infrastructure Consolidation
Logical Application Component	Service	Implements	Core
Logical Application Component	Logical Application Component	Decomposes	Core
Logical Application Component	Logical Application Component	Communicates with	Core
Logical Application Component	Logical Data Component	Used by	Data
Logical Application Component	Logical Technology Component	Is served by	Infrastructure Consolidation
Logical Application Component	Information System Service	Implements	Services
Logical Data Component	Logical Application Component	Uses	Data
Logical Data Component	Data Entity	Encapsulates	Data
Logical Data Component	Physical Data Component	Is realized by	Data
Logical Technology Component	Logical Application Component	Serves	Infrastructure Consolidation
Logical Technology Component	Physical Technology Component	Is realized by	Infrastructure Consolidation
Logical Technology Component	Technology Service	Supplies	Core
Logical Technology Component	Service	Provides platform for	Core
Logical Technology Component	Logical Technology Component	Decomposes	Core
Logical Technology Component	Logical Technology Component	Is dependent on	Core
Logical Technology Component	Logical Application Component	Serves	Infrastructure Consolidation

Source Entity	Target Entity	Name	Extension Module
Measure	Objective	Sets performance criteria for	Governance
Measure	Service	Sets performance criteria for	Governance
Measure	Measure	Decomposes	Governance
Objective	Goal	Realizes	Motivation
Objective	Measure	Is tracked against	Governance
Objective	Objective	Decomposes	Motivation
Organization Unit	Actor	Contains	Core
Organization Unit	Business Capability	Uses	Core
Organization Unit	Driver	Is motivated by	Motivation
Organization Unit	Function	Owns	Core
Organization Unit	Product	Produces	Process
Organization Unit	Process	Involves	Core
Organization Unit	Service	Owns and Governs	Core
Organization Unit	Organization Unit	Decomposes	Core
Physical Application Component	Logical Application Component	Realizes	Infrastructure Consolidation
Physical Application Component	Physical Data Component	Used by	Data
Physical Application Component	Physical Technology Component	Is served by	Core
Physical Application Component	Physical Application Component	Decomposes	Core
Physical Application Component	Physical Application Component	Communicates with	Core
Physical Data Component	Logical Data Component	Realizes	Data
Physical Data Component	Physical Data Component	Decomposes	Core
Physical Data Component	Physical Application Component	Uses	Data
Physical Technology Component	Physical Application Component	Serves	Core
Physical Technology Component	Logical Technology Component	Realizes	Infrastructure Consolidation
Physical Technology Component	Physical Technology Component	Decomposes	Core
Physical Technology Component	Physical Technology Component	Is dependent on	Core
Process	Control	Is guided by	Process
Process	Event	Generates	Process
Process	Event	Resolves	Process
Process	Function	Orchestrates	Core
Process	Function	Decomposes	Core
Process	Organization Unit	Involves	Core
Process	Product	Produces	Process
Process	Role	Involves	Core
Process	Role	Is performed by	Core
Process	Service	Orchestrates	Core
Process	Service	Decomposes	Core

Source Entity	Target Entity	Name	Extension Module
Process	Process	Decomposes	Core
Process	Process	Precedes/Follows	Core
Product	Organization Unit	Is produced by	Process
Product	Process	Is produced by	Process
Role	Actor	Is performed by	Core
Role	Process	Participates in	Core
Role	Process	Performs	Core
Role	Role	Decomposes	Core
Service	Actor	Is provided to	Core
Service	Contract	Is governed and measured by	Governance
Service	Data Entity	Provides	Core
Service	Data Entity	Consumes	Core
Service	Event	Resolves	Process
Service	Function	Provides governed interface to access	Core
Service	Information System Service	Is realized through	Services
Service	Logical Application Component	Is realized through	Core
Service	Logical Technology Component	Is implemented on	Core
Service	Measure	Is tracked against	Governance
Service	Organization Unit	Is owned and governed by	Core
Service	Process	Supports	Core
Service	Process	Is realized by	Core
Service	Service Quality	Meets	Governance
Service	Service	Consumes	Core
Service	Service	Decomposes	Core
Service Quality	Contract	Applies to	Governance
Service Quality	Service	Applies to	Governance
Technology Service	Logical Technology Component	Is supplied by	Core
Technology Service	Information System Service	Serves	Services
Value Stream	Actor	Involves	Core
Value Stream	Actor	Is triggered by	Core
Value Stream	Business Capability	Is enabled by	Core
Value Stream	Course of Action	Is influenced by	Core
Work Package	Capability	Delivers	Core

Chapter 31

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 Architecture Development Method (ADM).

31.1 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 31-1.[7]

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

7. Figure 31-1 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.

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.

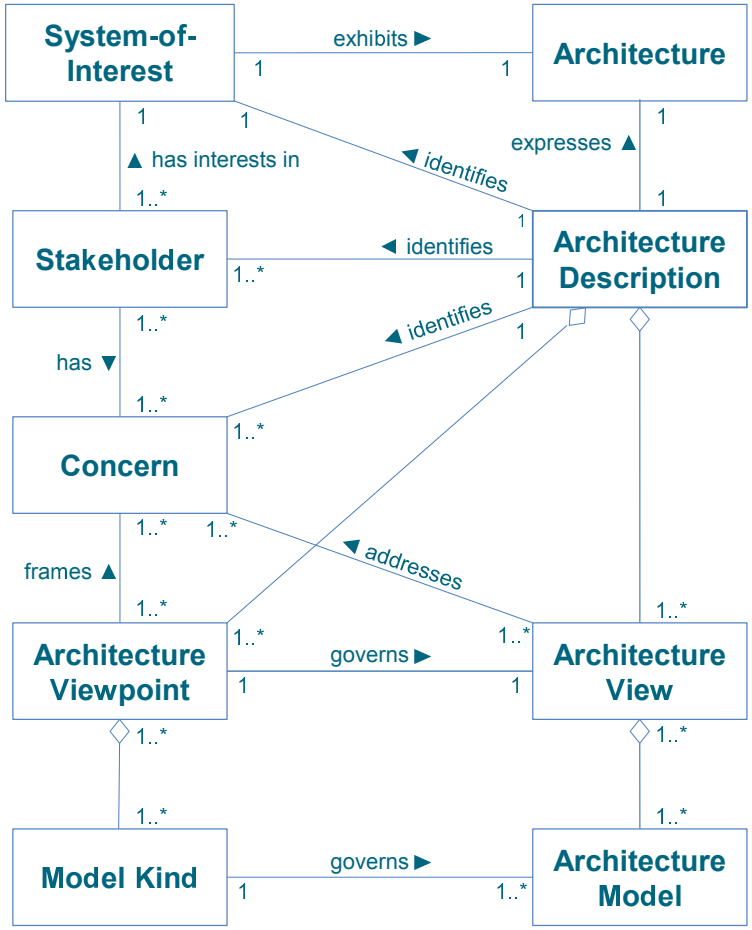

Figure 31-1 Basic Architectural Concepts

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.

Note: The terms "concern" and "requirement" are not synonymous. A concern is an area of interest. So, system reliability might be a concern/area of interest for some stakeholders. The reason why architects should identify concerns and associate them with architecture viewpoints, is to ensure that those concerns will be addressed in some fashion by the models of the architecture. For example, if the only architecture viewpoint selected by an architect is a structural architecture viewpoint, then reliability concerns are almost certainly not being addressed, since they cannot be represented in a structural model. Within that concern, stakeholders may have many distinct requirements: different classes of users may have very different reliability requirements for different capabilities of the system.

Concerns are the root of the process of decomposition into requirements. Concerns are represented in the architecture by these requirements. Requirements should be SMART (e.g., specific metrics).

31.1.1 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 31-2.

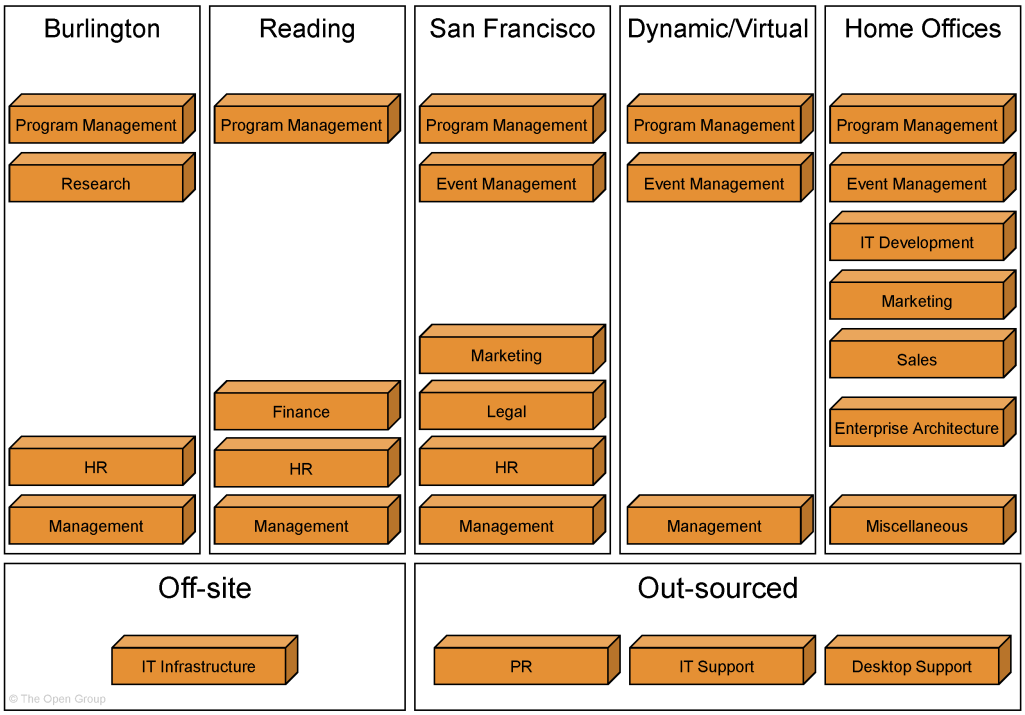

Figure 31-2 Example Architecture View — The Open Group Business Domains

31.2 Developing Architecture Views in the ADM

31.2.1 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 Part II: Architecture Development Method (ADM), 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 Part III, Chapter 23.

31.2.2 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?

31.3 Views, Tools, and Languages

The need for architecture views, and the process of developing them following the ADM, are 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.

31.3.1 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 Part V, Chapter 38.

31.4 Architecture Views and Architecture Viewpoints

31.4.1 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.

31.4.2 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.

31.4.3 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 Part V, Chapter 38.

31.5 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

31.6 Architectural Artifacts by ADM Phase

Catalog, Matrix, and Diagram Concept

The content 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 content metamodel.

The interactions between metamodel, building blocks, diagrams, and stakeholders are shown in Figure 31-3. Figure 31-4 shows the artifacts that are associated with the core content metamodel and each of the content extensions.

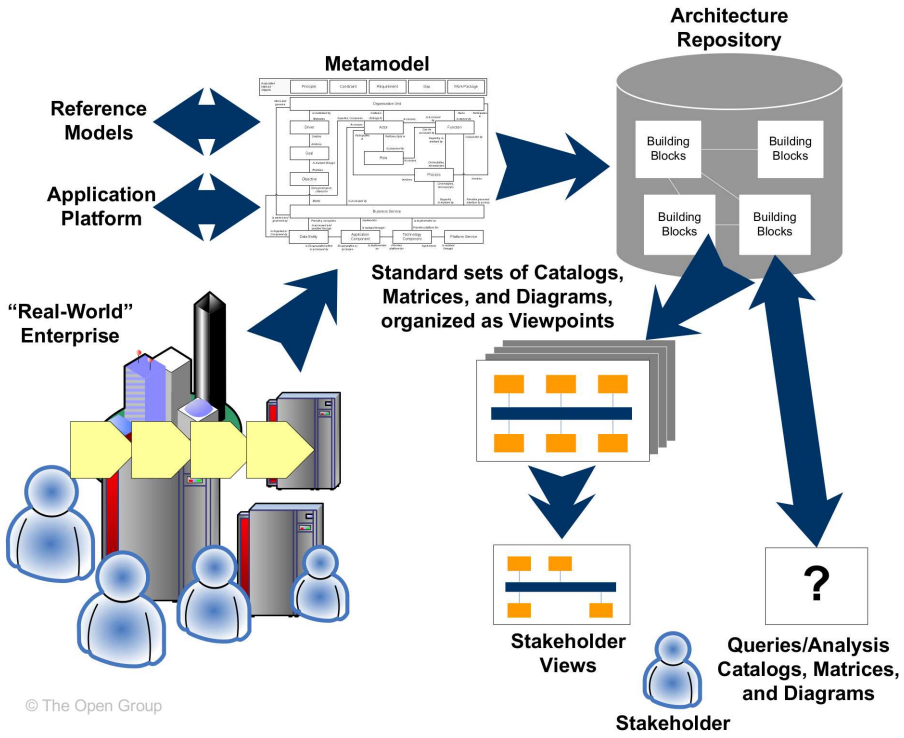

Figure 31-3 Interactions between Metamodel, Building Blocks, Diagrams, and Stakeholders

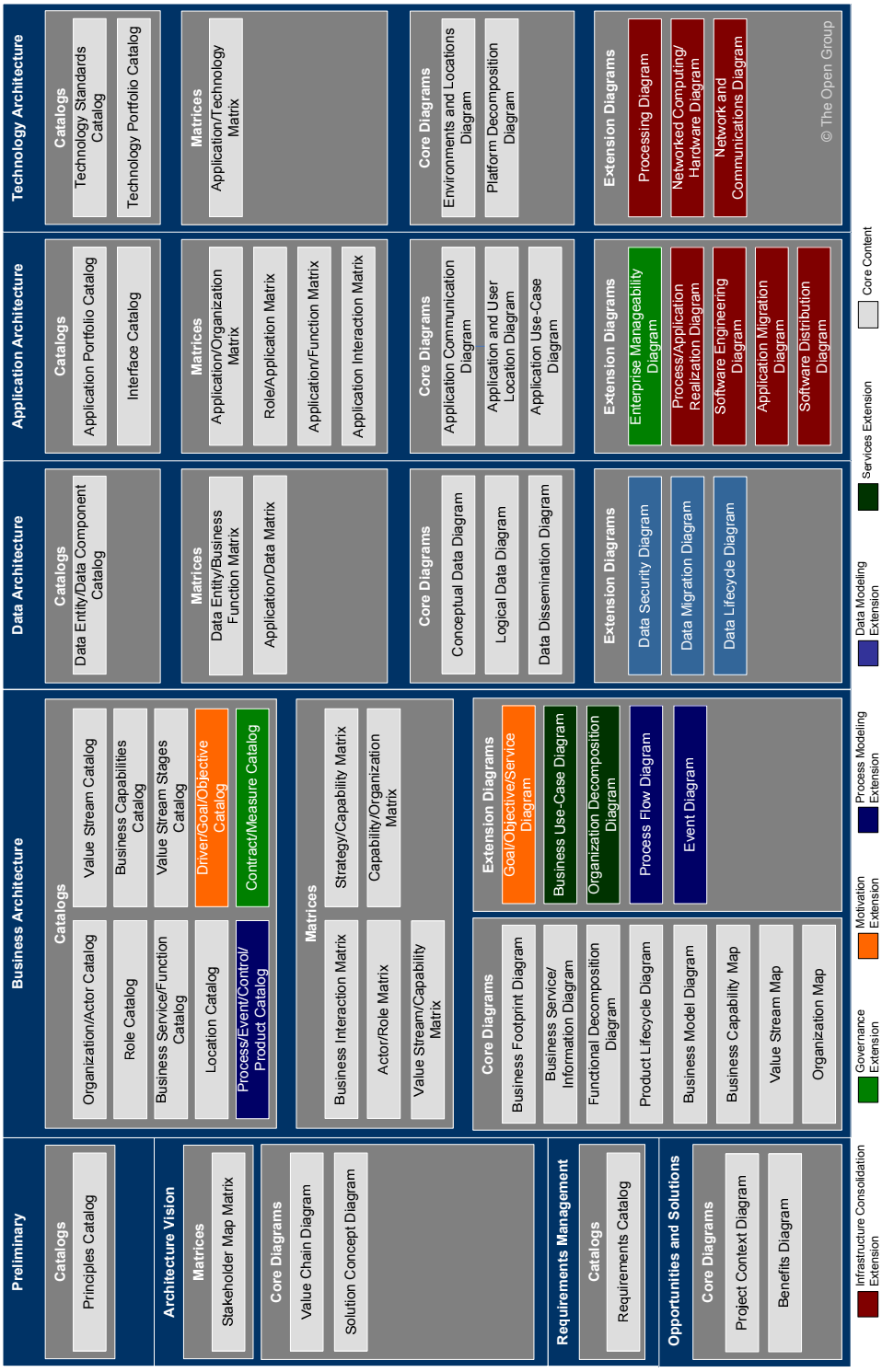

Figure 31-4 Artifacts Associated with the Core Content Metamodel and Extensions

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

31.6.1 Preliminary Phase

The following describes catalogs, matrices, and diagrams that may be created within the Preliminary Phase, as listed in Part II, Section 5.4.

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

31.6.2 Phase A: Architecture Vision

The following describes catalogs, matrices, and diagrams that may be created within Phase A (Architecture Vision) as listed in Section 6.4.

Stakeholder Map Matrix

The purpose of the Stakeholder Map matrix 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 Part III, Chapter 21).

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 map.

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 Functional Decomposition diagram 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 family of diagrams representing a definitive listing of a particular ability that a business may possess or exchange to achieve a specific purpose.

Value Stream Map

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

31.6.3 Phase B: Business Architecture

The following describes catalogs, matrices, and diagrams that may be created within Phase B (Business Architecture) as listed in Section 7.4.

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
- Information System 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
- Information System Service (optionally)
- Contract
- Measure

Business Capabilities Catalog

A definitive listing of particular abilities that a business may possess or exchange to achieve a specific purpose.

Value Stream Catalog

A definitive 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 definitive listing of end-to-end collections of the different stages for the value-adding activities that create an overall result for a customer, stakeholder, or end user; it includes the following metamodel entities:

- Business Capability
- Value Stream

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

The purpose of the Product Lifecycle diagram is to assist in understanding the lifecycles of key entities within the enterprise. Understanding product lifecycles is becoming increasingly important with respect to environmental concerns, legislation, and regulation where products must be tracked from manufacture to disposal. Equally, organizations that create products that involve personal or sensitive information must have a detailed understanding of the product lifecycle during the development of Business Architecture in order to ensure rigor in design of controls, processes, and procedures. Examples of this would include credit cards, debit cards, store/loyalty cards, smart cards, user identity credentials (identity cards, passports, etc.).

Goal/Objective/Service Diagram

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

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

Event Diagram

The purpose of the 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 family of diagrams representing a definitive listing of the particular abilities that a business may possess or exchange to achieve a specific purpose.

Value Stream Map

A family of diagrams representing a definitive listing of end-to-end collections 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.

31.6.4 Phase C: Data Architecture

The following describes catalogs, matrices, and diagrams that may be created within Phase C (Data Architecture) as listed in Section 9.4.

Data Entity/Data Component Catalog

The purpose of the Data Entity/Data Component catalog is to identify and maintain a list of all the data use across the enterprise, including data entities and also the data components where data entities are stored. An agreed Data Entity/Data Component catalog supports the definition and application of information management and data governance policies and also encourages effective data sharing and re-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 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

Data is considered as an asset to the enterprise and data security simply means ensuring that enterprise data is not compromised and that access to it is suitably controlled.

The purpose of the Data Security diagram is to depict which actor (person, organization, or system) can access which enterprise data. This relationship can be shown in a matrix form between two objects or can be shown as a mapping.

The diagram can also be used to demonstrate compliance with data privacy laws and other applicable regulations (HIPAA, Sarbanes-Oxley, etc). This diagram should also consider any trust implications where an enterprise's partners or other parties may have access to the company's systems, such as an outsourced situation where information may be managed by other people and may even be hosted in a different country.

Data Migration Diagram

Data migration is critical when implementing a package or packaged service-based solution. This is particularly true when an existing legacy application is replaced with a package or an enterprise is to be migrated to a larger package/packaged services footprint. Packages tend to have their own data model and during data migration the legacy application data may need to be transformed prior to loading into the package.

Data migration activities will usually involve the following steps:

- Extract data from source applications (baseline systems)
- Profile source data
- Perform data transformation operations, including data quality processes:
  - Standardize, normalize, de-duplicate source data (data cleansing)
  - Match, merge, and consolidate data from different source(s)
  - Source-to-target mappings
- Load into target applications (target systems)

The purpose of the Data Migration diagram is to show the flow of data from the source to the target applications. The diagram will provide a visual representation of the spread of sources/targets and serve as a tool for data auditing and establishing traceability. This diagram can be elaborated or enhanced as detailed as necessary. For example, the diagram can contain just an overall layout of migration landscape or could go into individual application metadata element level of detail.

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.

31.6.5 Phase C: Application Architecture

The following describes catalogs, matrices, and diagrams that may be created within Phase C (Application Architecture) as listed in Section 10.4.

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:

- Information System 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 functions are performed by organizational units. Some of the functions 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 functions
- 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.

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

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

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 the type of user licenses for the package or other software
- Estimation of the level of support needed for the users and location of 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 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.

31.6.6 Phase D: Technology Architecture

The following section describes catalogs, matrices, and diagrams that may be created within Phase D (Technology Architecture) as listed in Section 11.4.

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 lifecycle management of technology products and versions and also forms the basis for 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 details of specification, 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 and 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 grouping of business function, 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 UI 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 a 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.

31.6.7 Phase E: Opportunities and Solutions

The following section describes catalogs, matrices, and diagrams that may be created within Phase E (Opportunities & Solutions) as listed in Section 12.4.

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.

31.6.8 Requirements Management

The following section describes catalogs, matrices, and diagrams that may be created within the Requirements Management phase as listed in Section 16.4.

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., can the architecture meet all identified requirements).

The Requirements catalog contains the following metamodel entities:

- Requirement
- Assumption
- Constraint
- Gap

Chapter 32

Architecture Deliverables

This chapter provides descriptions of deliverables referenced in the Architecture Development Method (ADM).

32.1 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 Part IV, Chapter 29) 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 32.2.1)	F, H	A, B, C, D, E
Architecture Contract (see Section 32.2.2)	—	—
Architecture Definition Document (see Section 32.2.3)	B, C, D, E, F	C, D, E, F, G, H
Architecture Principles (see Section 32.2.4)	Preliminary, A, B, C, D	Preliminary, A, B, C, D, E, F, G, H
Architecture Repository (see Section 32.2.5)	Preliminary	Preliminary, A, B, C, D, E, F, G, H, Requirements Management
Architecture Requirements Specification (see Section 32.2.6)	B, C, D, E, F, Requirements Management	C, D, Requirements Management
Architecture Roadmap (see Section 32.2.7)	B, C, D, E, F	B, C, D, E, F
Architecture Vision (see Section 32.2.8)	A, E	B, C, D, E, F, G, H, Requirements Management

Deliverable	Output from...	Input to...
Business Principles, Business Goals, and Business Drivers (see Section 32.2.9)	Preliminary, A, B	A, B
Capability Assessment (see Section 32.2.10)	A, E	B, C, D, E, F
Change Request (see Section 32.2.11)	F, G, H	—
Communications Plan (see Section 32.2.12)	A	B, C, D, E, F
Compliance Assessment (see Section 32.2.13)	G	H
Implementation and Migration Plan (see Section 32.2.14)	E, F	F
Implementation Governance Model (see Section 32.2.15)	F	G, H
Organizational Model for Enterprise Architecture (see Section 32.2.16)	Preliminary	Preliminary, A, B, C, D, E, F, G, H, Requirements Management
Request for Architecture Work (see Section 32.2.17)	Preliminary, F, H	A, G
Requirements Impact Assessment (see Section 32.2.18)	Requirements Management	Requirements Management
Solution Building Blocks (see Section 32.2.19)	G	A, B, C, D, E, F, G
Statement of Architecture Work (see Section 32.2.20)	A, B, C, D, E, F, G, H	B, C, D, E, F, G, H, Requirements Management
Tailored Architecture Framework (see Section 32.2.21)	Preliminary, A	Preliminary, A, B, C, D, E, F, G, H, Requirements Management

32.2 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.

32.2.1 Architecture Building Blocks

Architecture documentation and models from the enterprise's Architecture Repository; see Part IV, Chapter 33.

32.2.2 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 Part VI, Chapter 44). 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 Part VI, Chapter 43.

32.2.3 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

32.2.4 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 Part III, Chapter 20 for guidelines and a detailed set of generic Architecture Principles, including:

- Business principles (see Section 20.6.1)
- Data principles (see Section 20.6.2)
- Application principles (see Section 20.6.3)
- Technology principles (see Section 20.6.4)

32.2.5 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 Part V, Chapter 37 for a detailed description of the content of an Architecture Repository.

32.2.6 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

32.2.7 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 Assessment and Deduction matrix, 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 (SBBs)

32.2.8 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 Version 0.1 Business, Application, Data, and Technology Architectures created; typically including:
  - Value Chain diagram
  - Solution Concept diagram
- Mapped requirements
- Reference to Draft Architecture Definition Document

32.2.9 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.

32.2.10 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

32.2.11 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

32.2.12 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 Critical Success Factor (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

32.2.13 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

32.2.14 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

32.2.15 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

32.2.16 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

32.2.17 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

32.2.18 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

32.2.19 Solution Building Blocks

Implementation-specific building blocks from the enterprise's Architecture Repository; see Part IV, Chapter 33.

32.2.20 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

32.2.21 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 Part II, Section 5.3.5 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 33

Building Blocks

This chapter explains the concept of building blocks.

33.1 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 33.2), discusses the general concepts of building blocks, and explains the differences between Architecture Building Blocks (ABBs) and Solution Building Blocks (SBBs)
- Building Blocks and the ADM (see Section 33.3), summarizes the stages at which building block design and specification occurs within the TOGAF Architecture Development Method (ADM)

33.2 Introduction to Building Blocks

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

33.2.1 Overview

This section describes the characteristics of building blocks. The use of building blocks in the ADM is described separately in Section 33.3.

33.2.2 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 has a type that corresponds to the enterprise's content 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 locks.

- 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 OMG has developed a standard for Re-usable Asset Specification (RAS),[8] which provides a good example of how building blocks can be formally described and managed.

33.2.3 Architecture Building Blocks

Architecture Building Blocks (ABBs) relate to the Architecture Continuum (see Part V, Section 35.4.1), and are defined or selected as a result of the application of the ADM.

33.2.3.1 Characteristics

ABBs:

- Capture architecture requirements; e.g., business, data, application, and technology requirements
- Direct and guide the development of SBBs

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

33.2.3.2 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

33.2.4 Solution Building Blocks

Solution Building Blocks (SBBs) relate to the Solutions Continuum (see Part V, Section 35.4.2), and may be either procured or developed.

33.2.4.1 Characteristics

SBBs:

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

33.2.4.2 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

33.3 Building Blocks and the ADM

33.3.1 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 33.2.

33.3.1.1 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

33.3.1.2 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.

33.3.2 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 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 parts of the ADM at which building blocks are designed and specified are summarized below.

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.

The specification of building blocks using the ADM is an evolutionary and iterative process. The key phases and steps of the ADM at which building blocks are evolved and specified are summarized below, and illustrated in Figure 33-1.

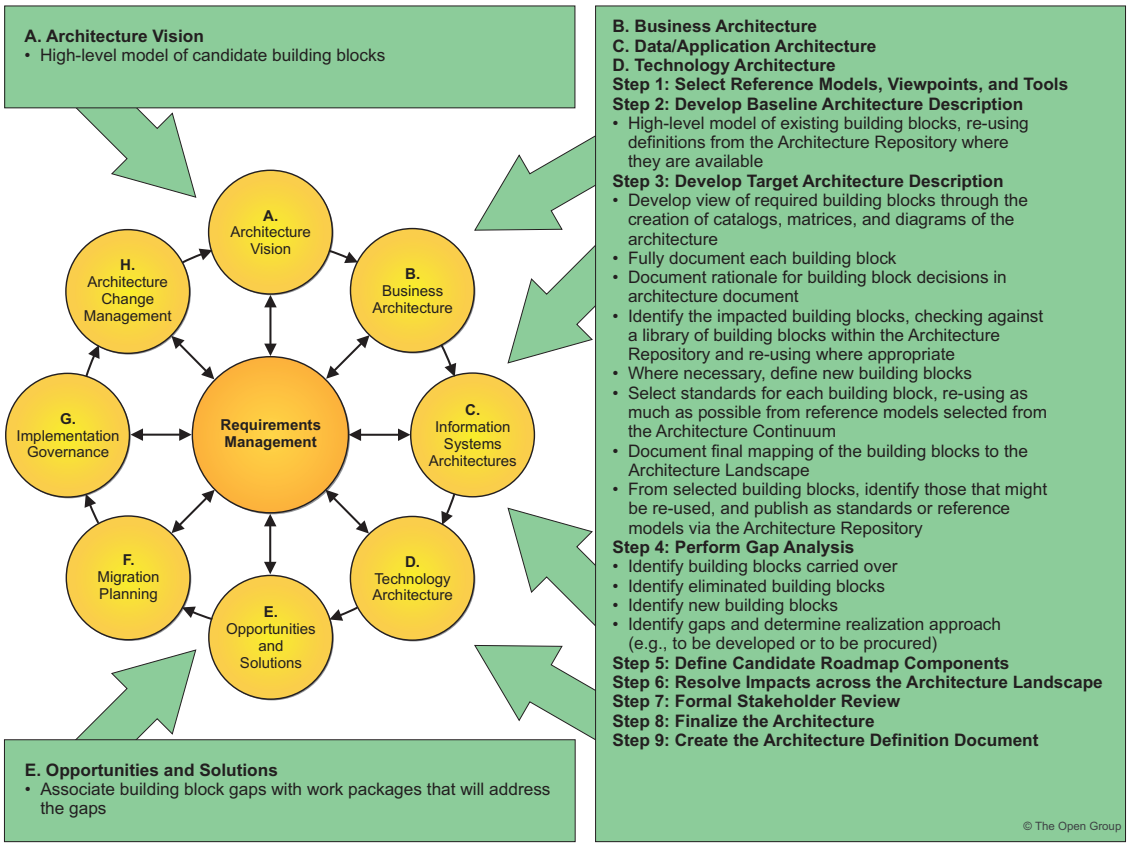

Figure 33-1 Key ADM Phases/Steps at which Building Blocks are Evolved/Specified

The TOGAF Standard, Version 9.2

Part V:

Enterprise Continuum and Tools

The Open Group

Chapter 34

Introduction to Part V

This chapter provides an introduction to and an overview of the contents of Part V: Enterprise Continuum & Tools.

34.1 Introduction

It is usually impossible to create a single unified architecture that meets all 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 will have a different purpose and architectures will 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.

This part of the TOGAF standard discusses the Enterprise Continuum; including the Architecture Continuum and the Solutions Continuum. It describes how architectures can be partitioned and organized within a repository. It also describes tools for architecture development.

34.2 Structure of Part V

Part V: Enterprise Continuum & Tools is structured as follows:

- Introduction (this chapter)
- The Enterprise Continuum (see Chapter 35) describes a view of the Architecture Repository that provides methods for classifying architecture and solution artifacts, showing how the different types of artifact evolve, and how they can be leveraged and re-used
- Architecture Partitioning (see Chapter 36) describes the various characteristics that can be applied to classify and then partition architectures
- The Architecture Repository (see Chapter 37) shows how the abstract classifications of architecture can be applied to a repository structure so that architectures can be organized and easily accessed
- Tools for Architecture Development (see Chapter 38) provides guidelines on selecting a toolset to create and manage architectural artifacts

Chapter 35

Enterprise Continuum

35.1 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 Commercial Off-The-Shelf (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.

35.2 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.

35.3 Constituents of the Enterprise Continuum

An overview of the context and constituents of the Enterprise Continuum is shown in Figure 35-1.

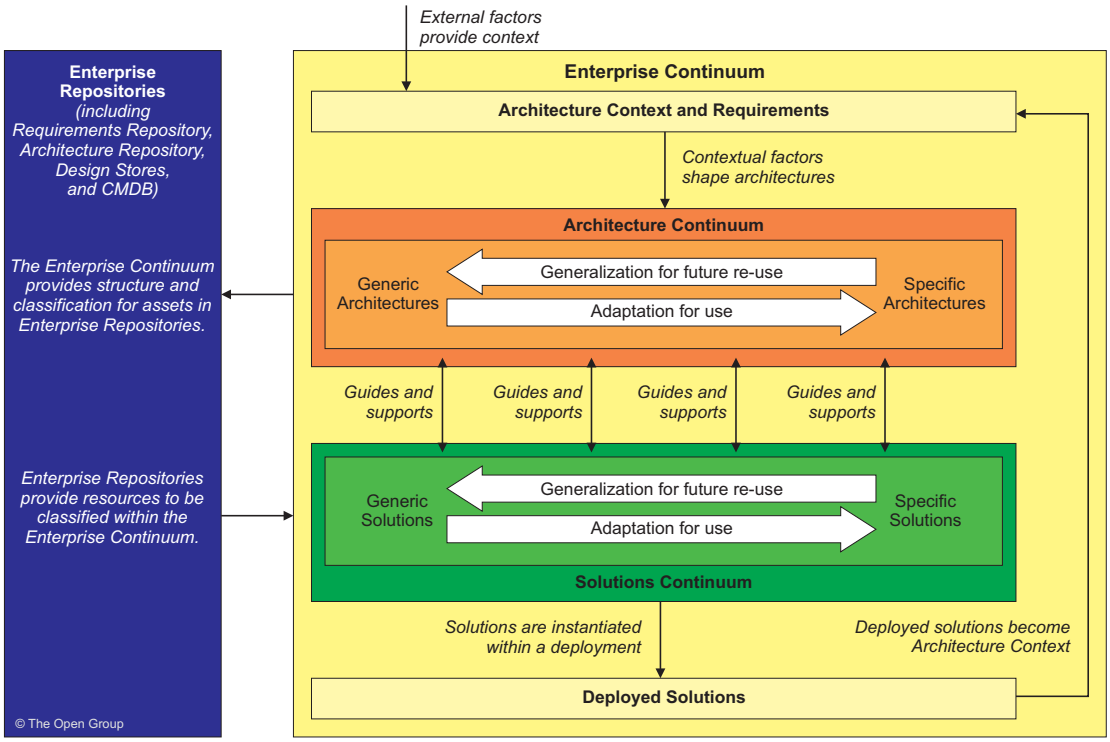

Figure 35-1 Enterprise Continuum

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

- The **Enterprise Continuum** (see Section 35.4) 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 35.4.1) 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 35.4.2) 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 Solution Building Blocks (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.

35.4 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.

35.4.1 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.

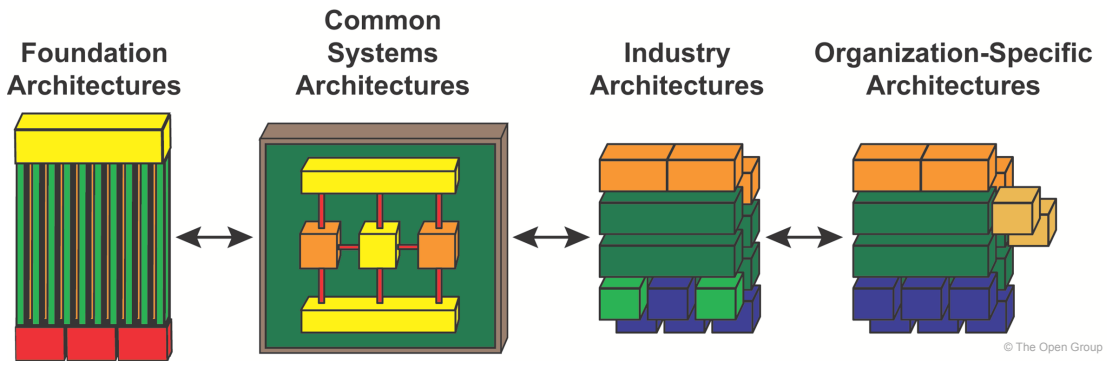

Figure 35-2 Architecture Continuum

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 35-2 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 35-2. 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 35-2 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 functions 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 Chapter 36.

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 architectural 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

35.4.2 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 35-3.

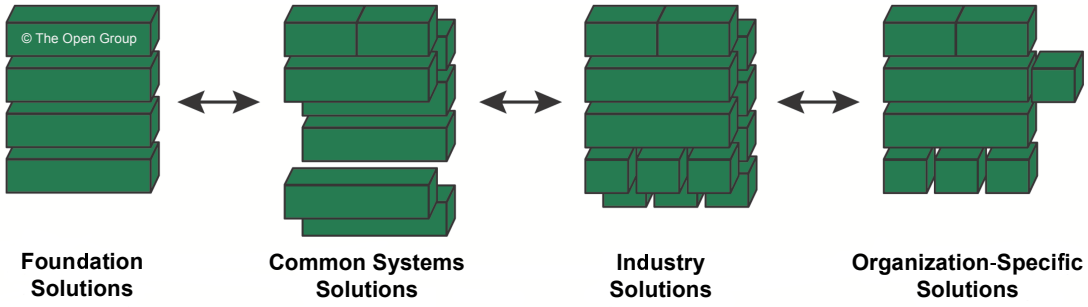

Figure 35-3 Solutions Continuum

"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 provides 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 functions. 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 Service-Level Agreements (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.

35.5 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.

35.6 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.

35.6.1 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 35-4.

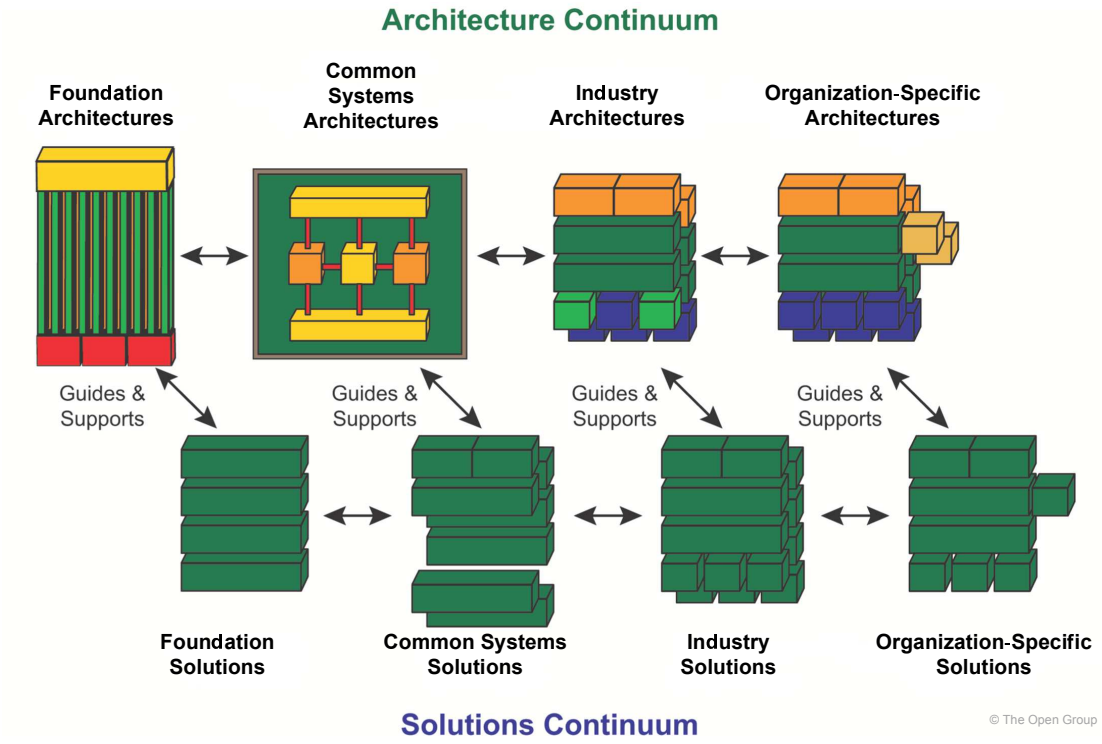

Figure 35-4 Relationships between Architecture and Solutions Continua

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 35-1 are an illustration showing opportunities for leveraging architecture and solution components.

35.6.2 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 36

Architecture Partitioning

36.1 Overview

Partitions are used to simplify the development and management of the Enterprise Architecture.

Partitions lie at the foundation of Architecture Governance and are distinct from levels and the organizing concepts of the Architecture Continuum (see Chapter 35).

Architectures are partitioned because:

- Organizational unit architectures conflict with one another
- Different teams need to work on different elements of architecture at the same time and partitions allow for specific groups of architects to own and develop specific elements of the architecture
- Effective architecture re-use requires modular architecture segments that can be taken and incorporated into broader architectures and solutions

It is impractical to present a definitive partitioning model for architecture. Each enterprise needs to adopt a partitioning model that reflects its own operating model.

This chapter discusses the classification criteria that are generally applied to architectures and how these can be leveraged to partition the enterprise into a set of architectures with manageable complexity and effective governance.

36.2 Applying Classification to Create Partitioned Architectures

For the reasons outlined in the previous section, it is valuable to partition and organize the Enterprise Continuum into a set of related solutions and architectures with:

- Manageable complexity for each individual architecture or solution
- Defined groupings
- Defined hierarchies and navigation structures
- Appropriate processes, roles, and responsibilities attached to each grouping

The following table shows how suitable classification criteria can be used to support partitioning of solutions:

Characteristic	Usage to Support Solution Partitioning
Subject Matter (Breadth)	Solutions are naturally organized into groups to support operational management and control. Examples of solution partitions according to subject matter would include applications, departments, divisions, products, services, service centers, sites, etc. Solution decomposition by subject matter is typically the fundamental technique for structuring both solutions and the architectures that represent them.
Time	Solution lifecycles are typically organized around a timeline, which allows the impact of solution development, introduction, operation, and retirement to be managed against other business activity occurring in similar time periods.
Maturity/Volatility	The maturity and volatility of a solution will typically impact the speed of execution required for the solution lifecycle. Additionally, volatility and maturity will shape investment priorities. Solutions existing in highly volatile environments may be better suited to rapid, agile development techniques.

The following table shows how each classification criteria can be used to support partitioning of architectures:

Characteristic	Usage to Support Architecture Partitioning
Depth	The level of detail within an architecture has a strong correlation to the stakeholder groups that will be interested in the architecture. Typically, less detailed architectures will be of interest to executive stakeholders. As architectures increase in detail, their relevance to implementation and operational personnel will also increase.

In practical terms, architecture discipline is used to support a number of different types of architecture that are used for different objectives. The classification criteria described above can be used in different ways to support the achievement of each objective.

The following characteristics are generally not used to partition an Architecture Landscape:

- Architectures used to describe the Architecture Landscape are generally not abstract
- Solution volatility generally prevents architectures from being defined that are far in the future; volatility also reduces the accuracy of historic architectures over time, as the organization changes and adapts to new circumstances

Using the criteria above, architectures can be grouped into partitions.

36.2.1 Activities within the Preliminary Phase

The key objective of the Preliminary Phase is to establish the Architecture Capability for the enterprise. In practical terms this activity will require the establishment of a number of architecture partitions, providing defined boundaries, governance, and ownership.

Generally speaking, each team carrying out architecture activity within the enterprise will own one or more architecture partitions and will execute the ADM to define, govern, and realize their architectures.

If more than one team is expected to work on a single architecture, this can become problematic, as the precise responsibilities of each team are difficult to establish. For this reason, it is preferable to apply partitioning to the architecture until each architecture has one owning team.

Finally, it is worth considering the distinction between standing capabilities of the enterprise and temporary teams mobilized to support a particular change initiative. Although the remit of standing teams within the enterprise can be precisely defined, it is more difficult to anticipate and specify the responsibilities of (possibly unknown) temporary architecture teams. In the cases of these temporary teams, each team should come under the governance of a standing architecture team and there should be a process within the ADM cycle of these teams to establish appropriate architecture partitioning.

Steps within the Preliminary Phase to support architecture partitioning are as follows:

- **Determine the organization structure for architecture within the enterprise**: the various standing teams that will create the architecture should be identified

  For each of these teams, appropriate boundaries should be established, including:

  — Governance bodies that are applicable to the team

  — Team membership

  — Team reporting lines

- **Determine the responsibilities for each standing architecture team**: for each architecture team, the responsibilities should be identified

  This step applies partitioning logic to the Enterprise Architecture in order to firstly identify the scope of each team and secondly to partition the architecture under the remit of a single team. Once complete, this step should have partitioned the entire scope of the enterprise and should have assigned responsibility for each partitioned architecture to a single team. Partitioning should create a definition of each architecture that includes:

  — Subject matter areas being covered

  — Level of detail at which the team will work

  — Time periods to be covered

  — Stakeholders

- **Determine the relationships between architectures**: once a set of partitioned architectures has been created, the relationships between architectures should be developed

  This step allows governance relationships to be formalized and also shows where artifacts from one architecture are expected to be re-used within other architectures. Areas of consideration include:

  — Where do different architectures overlap/dovetail/drill-down?

— What are the compliance requirements between architectures?

Once the Preliminary Phase is complete, the teams conducting the architecture should be understood. Each team should have a defined scope and the relationships between teams and architecture should be understood. Allocation of teams to architecture scope is illustrated in Figure 36-1.

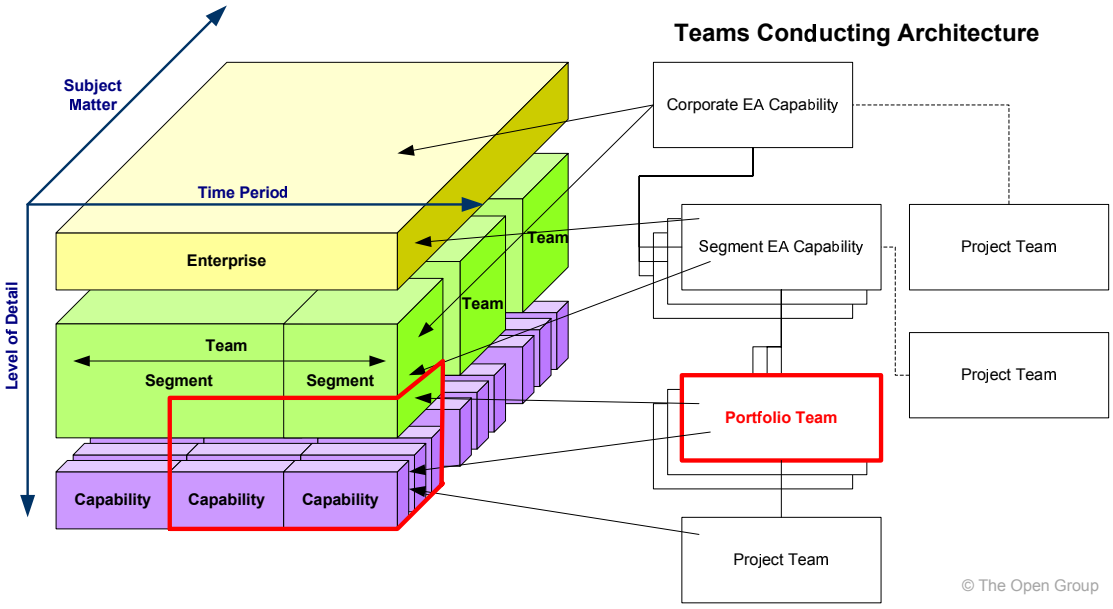

Figure 36-1 Allocation of Teams to Architecture Scope

36.3 Integration

Creation of partitioned architectures runs the risk of producing a fragmented and disjointed collection of architectures that cannot be integrated to form an overall big picture (see Part II, Section 4.6).

For large complex enterprises, federated architectures — independently developed, maintained, and managed architectures that are subsequently integrated within an integration framework — are typical. Federated architectures typically are used in governments and conglomerates, where the separate organizational units need separate architectures. Such a framework specifies the principles for interoperability, migration, and conformance. This allows specific business units to have architectures developed and governed as stand-alone architecture projects. More details and guidance on specifying the interoperability requirements for different solutions can be found in Part III, Chapter 25.

In order to mitigate against this risk, standards for content integration should be defined and Architecture Governance should address content integration as a condition of architectural compliance. Content frameworks, such as the TOGAF content framework (refer to Part IV: Architecture Content Framework) can be used to specify standard building blocks and artifacts that are the subject of content integration standards.

For example, a standard catalog of business processes can be agreed for an enterprise. Subsequent architectures can then ease integration by using the same process list and cross-referencing other aspects of the architecture to those standard processes.

Integration can be addressed from a number of dimensions:

- Integration across the architectural domains provides a cross-domain view of the state of a segment of the enterprise for a point in time
- Integration across the organizational scope of the business provides a cross-segment view of the enterprise
- The Architecture Vision provides an integrated summary of Architecture Definitions, which provide an integrated summary of Transition Architectures

Figure 36-2 shows how architectural content can be aggregated using a variety of techniques.

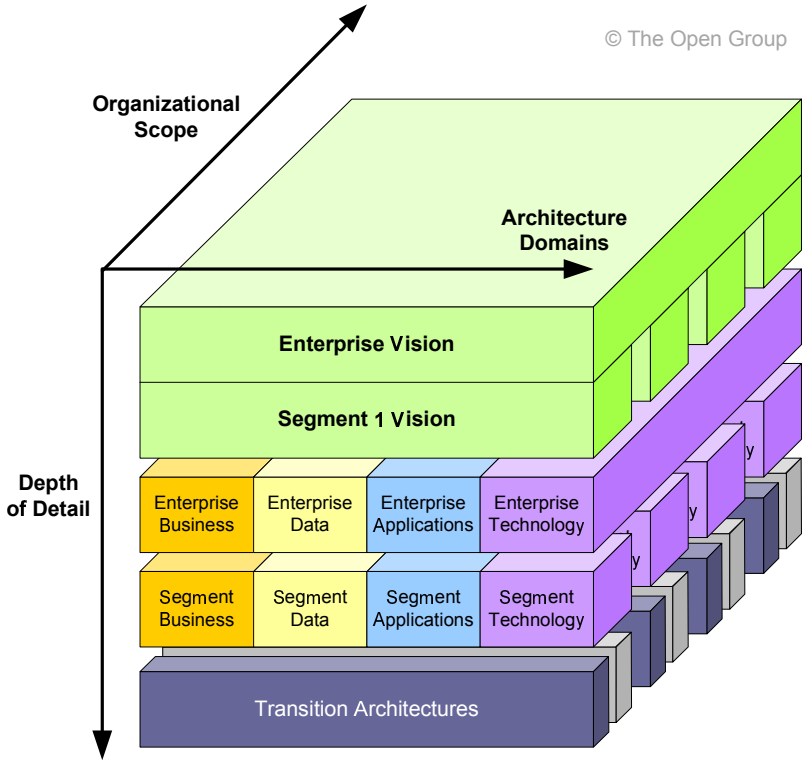

Figure 36-2 Architecture Content Aggregation

Chapter 37

Architecture Repository

37.1 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 Information Base** 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 Log** 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 Solution Building Blocks (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 37-1.

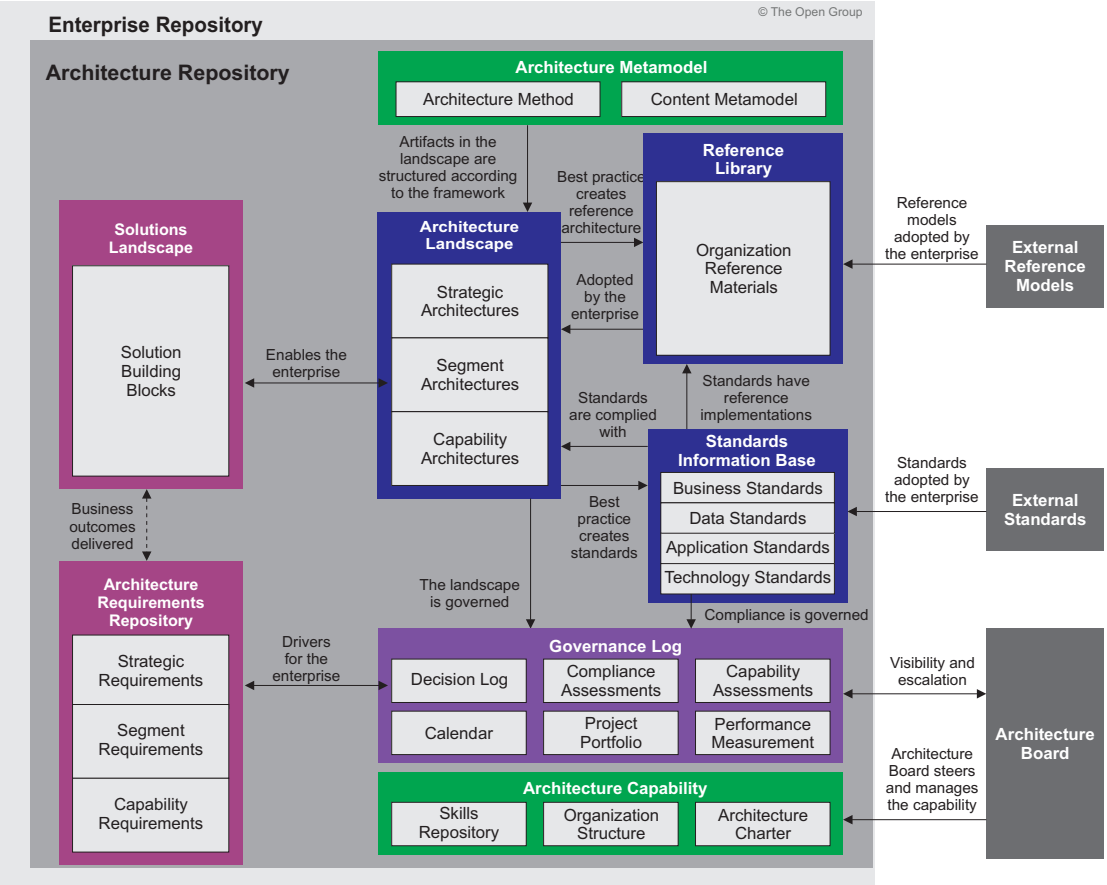

Figure 37-1 Overview of Architecture Repository

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

37.2 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 Part I, Section 3.74) 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 Part I, Section 3.64) 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 Part I, Section 3.31) 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.

37.3 Reference Library

37.3.1 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.

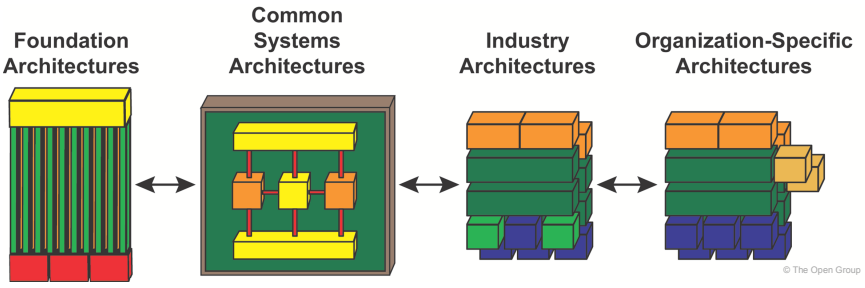

Figure 37-2 Architecture Continuum

The Architecture Continuum, as shown in Figure 37-2, 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 Chapter 36 and Part III, Chapter 19).

37.4 Standards Information Base

37.4.1 Overview

The Standards Information Base provides a repository area to hold a set of specifications, to which architectures must conform. Establishment of a Standards Information Base 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

37.4.2 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.

37.4.3 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.

37.4.4 Standards Classification within the Standards Information Base

Standards within the Standards Information Base are categorized according to the building blocks within the TOGAF content 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 functions
  - 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

37.5 Governance Log

37.5.1 Overview

The Governance Log 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.)

37.5.2 Contents of the Governance Log

The Governance Log 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.

37.6 The Architecture Requirements Repository

37.6.1 Overview

The Architecture Requirements Repository is used by all phases of the Architecture Development Method (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 lifecycle 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.

37.6.2 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 Enterprise Architecture lifecycle, 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.

37.7 Solutions Landscape

The Solutions Landscape holds the Solution Building Blocks (SBBs) which support the Architecture Building Blocks (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.

37.8 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.

37.9 External Repositories

37.9.1 External Reference Models

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

37.9.2 External Standards

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

37.9.3 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.

Chapter 38

Tools for Architecture Development

38.1 Overview

As an Enterprise Architecture framework, the TOGAF framework provides a basis for developing architectures in a uniform and consistent manner. Its purpose in this respect is to ensure that the various Architecture Descriptions developed within an enterprise, perhaps by different architects or architecture teams, support the comparison and integration of architectures within and across architecture domains (business, data, application, technology), and relating to different business area scopes within the enterprise.

To support this goal, the TOGAF standard defines numerous deliverables in the form of architectures, represented as architecture models, architecture views of those models, and other artifacts. Over time, these artifacts become a resource that needs to be managed and controlled, particularly with a view to re-use. This concept is referred to in the TOGAF standard as the "Enterprise Continuum".

Architecture models and views are discussed in detail separately in Part IV, Chapter 31. This section discusses considerations in choosing automated tools in order to generate such architecture models and views, and to maintain them over time.

38.2 Issues in Tool Standardization

In the current state of the tools market, many enterprises developing Enterprise Architectures struggle with the issue of standardizing on tools, whether they seek a single "one size fits all" tool or a multi-tool suite for modeling architectures and generating the different architecture views required.

There are ostensible advantages associated with selecting a single tool. Organizations following such a policy can hope to realize benefits such as reduced training, shared licenses, quantity discounts, maintenance, and easier data interchange. However, there are also reasons for refusing to identify a single mandated tool, including reasons of principle (endorsing a single architecture tool would not encourage competitive commercial innovation or the development of advanced tool capability); and the fact that a single tool would not accommodate a variety of architecture development "maturity levels" and specific needs across an enterprise.

Successful Enterprise Architecture teams are often those that harmonize their architecture tools with their architecture maturity level, team/organizational capabilities, and objectives or focus. If different organizations within an enterprise are at different architecture maturity levels and have different objectives or focus (e.g., Enterprise *versus* Business *versus* Technology Architecture), it becomes very difficult for one tool to satisfy all organizations' needs.

The TOGAF Standard, Version 9.2

Part VI:

Architecture Capability Framework

The Open Group

Chapter 39

Introduction to Part VI

This chapter provides an introduction to and an overview of the contents of Part VI: Architecture Capability Framework.

39.1 Overview

In order to successfully operate an architecture function within an enterprise, it is necessary to put in place appropriate organization structures, processes, roles, responsibilities, and skills to realize the Architecture Capability.

Part VI: Architecture Capability Framework provides a set of reference materials for how to establish such an architecture function. Readers should note that although this part contains a number of guidelines to support key activities, in its current form, the Architecture Capability Framework is not intended to be a comprehensive template for operating an Enterprise Architecture Capability.

An overall structure for the Architecture Capability Framework is shown in Figure 39-1.

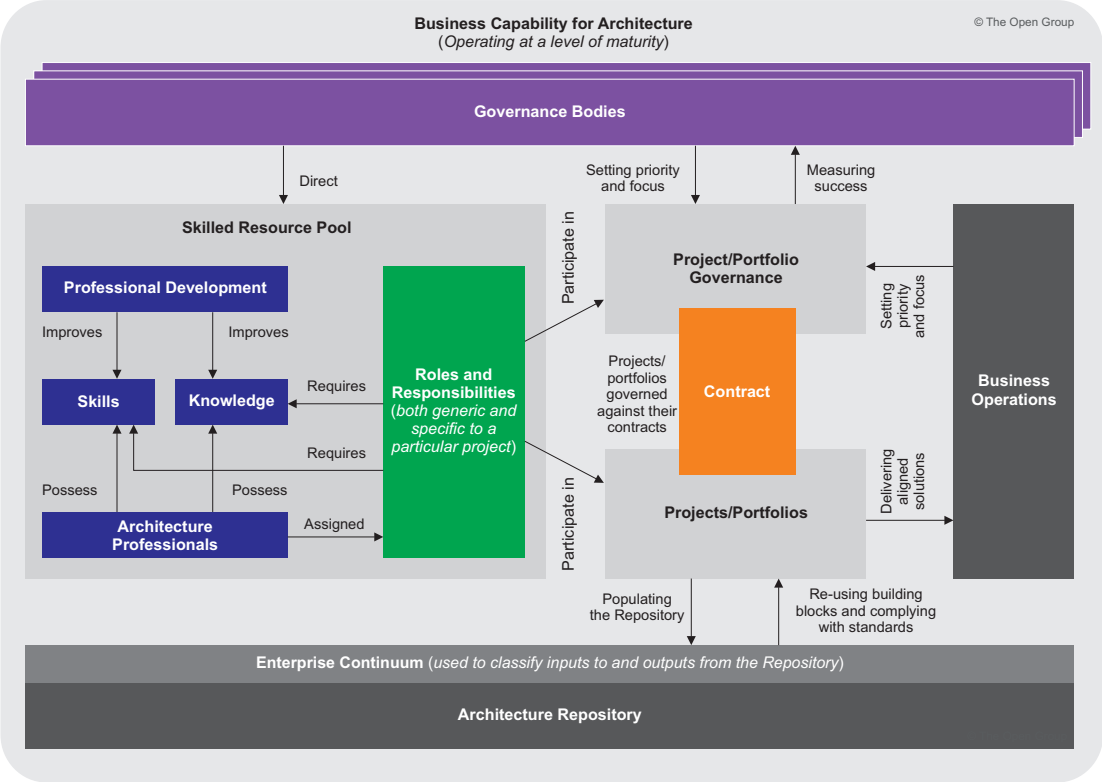

Figure 39-1 Mature Architecture Capability

39.2 Structure of Part VI

Part VI: Architecture Capability Framework is structured as follows:

- Introduction (this chapter)
- Establishing an Architecture Capability (see Chapter 40)
- Architecture Board (see Chapter 41)
- Architecture Compliance (see Chapter 42)
- Architecture Contracts (see Chapter 43)
- Architecture Governance (see Chapter 44)
- Architecture Maturity Models (see Chapter 45)
- Architecture Skills Framework (see Chapter 46)

Chapter 40

Establishing an Architecture Capability

This chapter provides guidelines on how to use the ADM to establish an Architecture Capability.

40.1 Overview

As with any business capability, the establishment of an Enterprise Architecture Capability can be supported by the TOGAF Architecture Development Method (ADM). Successful use of the ADM will provide a customer-focused, value-adding, and sustainable architecture practice that enables the business, helps maximize the value of investments, and pro-actively identifies opportunities to gain business benefits and manage risk.

Establishing a sustainable architecture practice within an organization can be achieved by adhering to the same approach that is used to establish any other capability — such as a Business Process Management (BPM) capability — within an organization. The ADM is an ideal method to be used to architect and govern the implementation of such a capability. Applying the ADM with the specific Architecture Vision to establish an architecture practice within the organization would achieve this objective.

This shouldn't be seen as a phase of an architecture project, or a one-off project, but rather as an ongoing practice that provides the context, environment, and resources to govern and enable architecture delivery to the organization. As an architecture project is executed within this environment it might request a change to the architecture practice that would trigger another cycle of the ADM to extend the architecture practice.

Implementing any capability within an organization would require the design of the four domain architectures: Business, Data, Application, and Technology. Establishing the architecture practice within an organization would therefore require the design of:

- The **Business Architecture** of the architecture practice that will highlight the Architecture Governance, architecture processes, architecture organizational structure, architecture information requirements, architecture products, etc.
- The **Data Architecture** that would define the structure of the organization's Enterprise Continuum and Architecture Repository
- The **Application Architecture** specifying the functionality and/or applications services required to enable the architecture practice
- The **Technology Architecture** that depicts the architecture practice's infrastructure requirements and deployment in support of the architecture applications and Enterprise Continuum

The steps in establishing an architecture practice are explained below, against the context of the ADM phases. The reader should therefore refer to the relevant ADM phase in Part II: Architecture Development Method (ADM) to understand the complete scope of each step. In this section, key aspects will be highlighted for each ADM phase that should be considered and

are specific to establishing an architecture practice. The intent is therefore not to repeat each ADM phase description, but to guide the reader to apply each ADM phase within the context of establishing an architecture practice.

40.2 Phase A: Architecture Vision

The purpose of this phase within the context of establishing an architecture practice is to define or review the vision, stakeholders, and principles of the architecture practice. The focus in this phase would be on the architecture practice as a whole and not on a particular architecture project.

The following should be considered in terms of understanding the steps in the context of establishing an architecture practice:

- **Establish the Project**: this step should focus on defining the stakeholders in the architecture practice

  The stakeholders would include the roles and organization units participating in the architecture practice, as well as those that will benefit from the deliverables generated by the architecture practice that can therefore be defined as customers of the architecture practice.

- **Identify Stakeholders and Concerns, Business Requirements, and Architecture Vision**: this step generates the first, very high-level definitions of the baseline and target environments, from a business information systems and technology perspective for the architecture practice

- **Identify Business Goals and Business Drivers**: this would be more relevant for the architecture practice than for a particular architecture project; an understanding of the business goals and drivers is essential to align the architecture practice to the business

- **Define Scope**: defining the scope of the architecture practice would be a high-level project plan of what should be addressed in terms of architecture for the next period

- **Define Constraints**: the focus in this step should be on the enterprise-wide constraints that would impact on all architecture projects

- **Review Architecture Principles, including Business Principles**: the intent in this step should be to define the principles that would govern and guide the running of the architecture practice

  Where Architecture Principles usually govern the architecture deliverables, the architecture practice principles would address the architecture practice organization, content, tools, and process.

- **Develop Statement of Architecture Work and Secure Approval**: this step should generate the architecture practice vision and scope

Another step that can be considered during this phase is to conduct an architecture maturity assessment. Refer to Chapter 45 for guidance on this topic.

40.3 Phase B: Business Architecture

Key areas of focus during this phase of establishing or refining the Business Architecture of the architecture practice are:

- An **Architecture Ontology** defining the architectural terms and definitions that will be used in the organization in order to establish a common understanding of these terms
- The **Architecture Process** where the ADM would form the base of the process and need to be customized to meet the organization's requirements and architecture practice vision

  Refer to Section 4.3 for guidance on developing this process. The required Architecture Governance processes should be included in the overall architecture process.
- The **Architecture Viewpoints and Views** that list all the viewpoints and views that should be addressed by the architecture practice

  The identified architecture practice stakeholders would guide the development of this definition. One of the viewpoints to be included is the Architecture Governance viewpoint; refer to Part IV, Chapter 31 for guidance on this output.
- The **Architecture Framework** describing the various architecture deliverables that will be generated by the architecture practice, the inter-relationships and dependencies between the architecture deliverables, as well as the rules and guidelines governing the design of these deliverables

  The defined architecture viewpoints and views should be used to guide the definition of the architecture framework. Part II: Architecture Development Method (ADM) and Chapter 32 are useful references that will assist in describing the architecture framework.
- The **Architecture Accountability Matrix** defining the roles in the architecture practice and allocating accountability of the roles to architecture deliverables and processes

  This matrix would include the required Architecture Governance structures and roles. Part II: Architecture Development Method (ADM) as well as Chapter 41, Chapter 44, and Chapter 46 would provide guidance on this output.
- The **Architecture Performance Metrics** identifying and describing the metrics that will be used to monitor the performance of the architecture practice against its stated architecture practice vision and objectives
- The **Architecture Governance Framework** which is a specific view of the defined architecture process and Architecture Accountability Matrix

40.4 Phase C: Data Architecture

The Data Architecture of the architecture practice would specify and govern the structure of the organization's Enterprise Continuum and Architecture Repository. The Data Architecture should be defined based on the architecture framework. The Data Architecture is sometimes referred to as the metamodel of the architecture practice.

40.5 Phase C: Application Architecture

The Application Architecture of the architecture practice defines the functionality required to generate, maintain, publish, distribute, and govern the architecture deliverables as defined in the architecture framework. A key focus should be on the modeling toolsets required for modeling, but it should not be the only focus. Refer to Chapter 38 for guidance on selecting a toolset. Publishing the architecture deliverables to address specific views in the architecture framework would sometimes require specialized or customized functionality and should not be neglected.

40.6 Phase D: Technology Architecture

The Technology Architecture of the architecture practice should define technology infrastructure supporting the architecture practice.

40.7 Phase E: Opportunities & Solutions

A critical factor to consider during this phase of planning the establishment of the architecture practice is the organizational change that is required and how this will be achieved.

40.8 Phase F: Migration Planning

The focus should not only be on the Information Systems Architecture components in this phase, but include the Business Architecture. The adoption of the architecture process and framework will have a major impact on the overall establishment of the architecture practice in the organization.

40.9 Phase G: Implementation Governance

The implementation of the Business Architecture of the architecture practice should be the focus of this phase. Changing practices within the organization to adopt a more structured and disciplined approach will be a challenge and should be addressed by the appropriate organizational change techniques.

40.10 Phase H: Architecture Change Management

Changes to the architecture of the architecture practice should be managed by this phase. These changes are usually triggered during the execution of architecture projects. A typical change would be the requirement for a new architecture deliverable. This would impact on all the architecture domains of the architecture practice.

40.11 Requirements Management

Understanding and managing the requirements for the architecture practice is crucial. Requirements should be clearly articulated and align to the architecture practice vision.

Chapter 41

Architecture Board

This chapter provides guidelines for establishing and operating an Enterprise Architecture Board.

41.1 Role

A key element in a successful Architecture Governance strategy (see Chapter 44) is a cross-organization Architecture Board to oversee the implementation of the strategy. This body should be representative of all the key stakeholders in the architecture, and will typically comprise a group of executives responsible for the review and maintenance of the overall architecture.

Architecture Boards may have global, regional, or business line scope. Particularly in larger enterprises, Architecture Boards typically comprise representatives from the organization at a minimum of two levels:

- Local (domain experts, line responsibility)
- Global (organization-wide responsibility)

In such cases, each board will be established with identifiable and articulated:

- Responsibilities and decision-making capabilities
- Remit and authority limits

41.2 Responsibilities

The Architecture Board is typically made responsible, and accountable, for achieving some or all of the following goals:

- Providing the basis for all decision-making with regard to the architectures
- Consistency between sub-architectures
- Establishing targets for re-use of components
- Flexibility of the Enterprise Architecture:
  - To meet changing business needs
  - To leverage new technologies
- Enforcement of Architecture Compliance
- Improving the maturity level of architecture discipline within the organization
- Ensuring that the discipline of architecture-based development is adopted

- Supporting a visible escalation capability for out-of-bounds decisions

Further responsibilities from an operational perspective should include:

- All aspects of monitoring and control of the Architecture Contract
- Meeting on a regular basis
- Ensuring the effective and consistent management and implementation of the architectures
- Resolving ambiguities, issues, or conflicts that have been escalated
- Providing advice, guidance, and information
- Ensuring compliance with the architectures, and granting dispensations that are in keeping with the technology strategy and objectives
- Considering policy (schedule, Service-Level Agreements (SLAs), etc.) changes where similar dispensations are requested and granted; e.g., new form of service requirement
- Ensuring that all information relevant to the implementation of the Architecture Contract is published under controlled conditions and made available to authorized parties
- Validation of reported service levels, cost savings, etc.

From a governance perspective, the Architecture Board is also responsible for:

- The production of usable governance material and activities
- Providing a mechanism for the formal acceptance and approval of architecture through consensus and authorized publication
- Providing a fundamental control mechanism for ensuring the effective implementation of the architecture
- Establishing and maintaining the link between the implementation of the architecture, the architectural strategy and objectives embodied in the Enterprise Architecture, and the strategic objectives of the business
- Identifying divergence from the architecture and planning activities for realignment through dispensations or policy updates

41.3 Setting Up the Architecture Board

41.3.1 Triggers

One or more of the following occurrences typically triggers the establishment of an Architecture Board:

- New CIO
- Merger or acquisition
- Consideration of a move to newer forms of computing
- Recognition that IT is poorly aligned to business
- Desire to achieve competitive advantage via technology
- Creation of an Enterprise Architecture program

- Significant business change or rapid growth
- Requirement for complex, cross-functional solutions

In many companies, the executive sponsor of the initial architecture effort is the CIO (or other senior executive). However, to gain broad corporate support, a sponsoring body has more influence. This sponsoring body is here called an Architecture Board, but the title is not important. Whatever the name, it is the executive-level group responsible for the review and maintenance of the strategic architecture and all of its sub-architectures.

The Architecture Board is the sponsor of the architecture within the enterprise, but the Architecture Board itself needs an executive sponsor from the highest level of the corporation. This commitment must span the planning process and continue into the maintenance phase of the architecture project. In many companies that fail in an architecture planning effort, there is a notable lack of executive participation and encouragement for the project.

A frequently overlooked source of Architecture Board members is the company's Board of Directors. These individuals invariably have diverse knowledge about the business and its competition. Because they have a significant impact on the business vision and objectives, they may be successful in validating the alignment of IT strategies to business objectives.

41.3.2 Size of the Board

The recommended size for an Architecture Board is four or five (and no more than ten) permanent members. In order to keep the Architecture Board to a reasonable size, while ensuring enterprise-wide representation on it over time, membership of the Architecture Board may be rotated, giving decision-making privileges and responsibilities to various senior managers. This may be required in any case, due to some Architecture Board members finding that time constraints prevent long-term active participation.

However, some continuity must exist on the Architecture Board, to prevent the corporate architecture from varying from one set of ideas to another. One technique for ensuring rotation with continuity is to have set terms for the members, and to have the terms expire at different times.

In the ongoing architecture process following the initial architecture effort, the Architecture Board may be re-chartered. The executive sponsor will normally review the work of the Architecture Board and evaluate its effectiveness; if necessary, the Architecture Compliance review process is updated or changed.

41.3.3 Board Structure

The TOGAF Architecture Governance Framework (see Section 44.2) provides a generic organizational framework that positions the Architecture Board in the context of the broader governance structures of the enterprise. This structure identifies the major organizational groups and responsibilities, as well as the relationship between each group. This is a best practice structure, and may be subject to change depending on the organization's form and existing structures.

Consideration must be given to the size of the organization, its form, and how the IT functions are implemented. This will provide the basis for designing the Architecture Board structure within the context of the overall governance environment. In particular, consideration should be given to the concept of global ownership and local implementation, and the integration of new concepts and technologies from all areas implementing against architectures.

The structure of the Architecture Board should reflect the form of the organization. The Architecture Governance structure required may well go beyond the generic structures outlined in the TOGAF Architecture Governance Framework (see Section 44.2). The organization may need to define a combination of the IT governance process in place and the existing organizational structures and capabilities, which typically include the following types of body:

- Global governance board
- Local governance board
- Design authorities
- Working parties

41.4 Operation of the Architecture Board

This section describes the operation of the Architecture Board particularly from the governance perspective.

41.4.1 General

Architecture Board meetings should be conducted within clearly identified agendas with explicit objectives, content coverage, and defined actions. In general, board meetings will be aligned with best practice, such as given in the COBIT framework (see Section 44.1.4.1).

These meetings will provide key direction in:

- Supporting the production of quality governance material and activities
- Providing a mechanism for formal acceptance through consensus and authorized publication
- Providing a fundamental control mechanism for ensuring the effective implementation of the architectures
- Establishing and maintaining the link between the implementation of the architectures and the stated strategy and objectives of the organization (business and IT)
- Identifying divergence from the contract and planning activities to realign with the contract through dispensations or policy updates

41.4.2 Preparation

Each participant will receive an agenda and any supporting documentation — e.g., dispensation requests, performance management reports, etc. — and will be expected to be familiar with the contents of each.

Where actions have been allocated to an individual, it is that person's responsibility to report on progress against these.

Each participant must confirm their availability and attendance at the Architecture Board meeting.

41.4.3 Agenda

This section outlines the contents of an Architecture Board meeting agenda. Each agenda item is described in terms of its content only.

Minutes of Previous Meeting

Minutes contain the details of previous Architecture Board meetings as per standard organizational protocol.

Requests for Change

Items under this heading are normally change requests for amendments to architectures, principles, etc., but may also include business control with regard to Architecture Contracts; e.g., ensure that voice traffic to premium numbers, such as weather reports, is barred and data traffic to certain websites is controlled.

Any request for change is made within agreed authority levels and parameters defined by the Architecture Contract.

Dispensations

A dispensation is used as the mechanism to request a change to the existing architectures, contracts, principles, etc. outside of normal operating parameters; e.g., exclude provision of service to a subsidiary, request for unusual service levels for specific business reasons, deploy non-standard technology or products to support specific business initiatives.

Dispensations are granted for a given time period and set of identified services and operational criteria that must be enforced during the lifespan of the dispensation. Dispensations are not granted indefinitely, but are used as a mechanism to ensure that service levels and operational levels, etc. are met while providing a level flexibility in their implementation and timing. The time-bound nature of dispensations ensures that they are a trigger to the Architecture Compliance activity.

Compliance Assessments

Compliance is assessed against SLAs, Operational-Level Agreements (OLAs), cost targets, and required architecture refreshes. These assessments will be reviewed and either accepted or rejected depending on the criteria defined within the Architecture Governance Framework. The Architecture Compliance assessment report will include details as described.

Dispute Resolution

Disputes that have not been resolved through the Architecture Compliance and dispensation processes are identified here for further action and are documented through the Architecture Compliance assessments and dispensation documentation.

Architecture Strategy and Direction Documentation

This describes the architecture strategies, direction, and priorities and will only be formulated by the global Architecture Board. It should take the form of standard architecture documentation.

Actions Assigned

This is a report on the actions assigned at previous Architecture Board meetings. An action tracker is used to document and keep the status of all actions assigned during the Architecture Board meetings and should consist of at least the following information:

- Reference
- Priority
- Action description
- Action owner
- Action details
- Date raised
- Due date
- Status
- Type
- Resolution date

Contract Documentation Management

This is a formal acceptance of updates and changes to architecture documentation for onward publication.

Any Other Business (AOB)

Description of issues not directly covered under any of the above. These may not be described in the agenda but should be raised at the beginning of the meeting. Any supporting documentation must be managed as per all Architecture Governance documentation.

Schedule of Meetings

All meeting dates should be detailed and published.

Chapter 42

Architecture Compliance

This chapter provides guidelines for ensuring project compliance to the architecture.

42.1 Introduction

Ensuring the compliance of individual projects with the Enterprise Architecture is an essential aspect of Architecture Governance (see Chapter 44). To this end, the IT governance function within an enterprise will normally define two complementary processes:

- The **Architecture** function will be required to prepare a series of Project Architectures; i.e., project-specific views of the Enterprise Architecture that illustrate how the Enterprise Architecture impacts on the major projects within the organization (see ADM Phases A to F)
- The **IT Governance** function will define a formal Architecture Compliance review process (see Section 42.3) for reviewing the compliance of projects to the Enterprise Architecture

Apart from defining formal processes, the Architecture Governance function (see Chapter 44) may also stipulate that the architecture function should extend beyond the role of architecture definition and standards selection, and participate also in the technology selection process, and even in the commercial relationships involved in external service provision and product purchases. This may help to minimize the opportunity for misinterpretation of the Enterprise Architecture, and maximize the value of centralized commercial negotiation.

42.2 Terminology: The Meaning of Architecture Compliance

A key relationship between the architecture and the implementation lies in the definitions of the terms "conformant", "compliant", etc. While terminology usage may differ between organizations, the concepts of levels of conformance illustrated in Figure 42-1 should prove useful in formulating an IT compliance strategy.

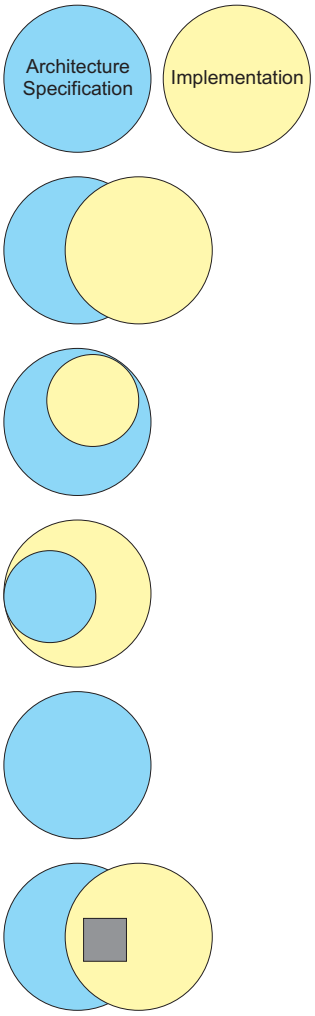

Figure 42-1 Levels of Architecture Conformance

The phrase "in accordance with" in Figure 42-1 means:

- Supports the stated strategy and future directions
- Adheres to the stated standards (including syntax and semantic rules specified)
- Provides the stated functionality
- Adheres to the stated principles; for example:
  - Open wherever possible and appropriate
  - Re-use of component building blocks wherever possible and appropriate

42.3 Architecture Compliance Reviews

An Architecture Compliance review is a scrutiny of the compliance of a specific project against established architectural criteria, spirit, and business objectives. A formal process for such reviews normally forms the core of an Enterprise Architecture Compliance strategy.

42.3.1 Purpose

The goals of an Architecture Compliance review include some or all of the following:

- First and foremost, catch errors in the project architecture early, and thereby reduce the cost and risk of changes required later in the lifecycle

  This in turn means that the overall project time is shortened, and that the business gets the bottom-line benefit of the architecture development faster.
- Ensure the application of best practices to architecture work
- Provide an overview of the compliance of an architecture to mandated enterprise standards
- Identify where the standards themselves may require modification
- Identify services that are currently application-specific but might be provided as part of the enterprise infrastructure
- Document strategies for collaboration, resource sharing, and other synergies across multiple architecture teams
- Take advantage of advances in technology
- Communicate to management the status of technical readiness of the project
- Identify key criteria for procurement activities (e.g., for inclusion in Commercial Off-The-Shelf (COTS) product RFI/RFP documents)
- Identify and communicate significant architectural gaps to product and service providers

Apart from the generic goals related to quality assurance outlined above, there are additional, more politically-oriented motivations for conducting Architecture Compliance reviews, which may be relevant in particular cases:

- The Architecture Compliance review can be a good way of deciding between architectural alternatives, since the business decision-makers typically involved in the review can guide decisions in terms of what is best for the business, as opposed to what is technically more pleasing or elegant
- The output of the Architecture Compliance review is one of the few measurable deliverables to the CIO to assist in decision-making
- Architecture reviews can serve as a way for the architecture organization to engage with development projects that might otherwise proceed without involvement of the architecture function
- Architecture reviews can demonstrate rapid and positive support to the enterprise business community:
  - The Enterprise Architecture and Architecture Compliance helps ensure the alignment of IT projects with business objectives

— Architects can sometimes be regarded as being deep into technical infrastructure and far removed from the core business

— Since an Architecture Compliance review tends to look primarily at the critical risk areas of a system, it often highlights the main risks for system owners

While compliance to architecture is required for development and implementation, non-compliance also provides a mechanism for highlighting:

- Areas to be addressed for realignment
- Areas for consideration for integration into the architectures as they are uncovered by the compliance processes

The latter point identifies the ongoing change and adaptability of the architectures to requirements that may be driven by indiscipline, but also allows for changes to be registered by faster moving changes in the operational environment. Typically, dispensations (see Section 44.1.4) will be used to highlight these changes and set in motion a process for registering, monitoring, and assessing the suitability of any changes required.

42.3.2 Timing

Timing of compliance activities should be considered with regard to the development of the architectures themselves.

Compliance reviews are held at appropriate project milestones or checkpoints in the project's lifecycle. Specific checkpoints should be included as follows:

- Development of the architecture itself (ADM compliance)
- Implementation of the architecture(s) (architecture compliance)

Architecture project timings for assessments should include:

- Project initiation
- Initial design
- Major design changes
- *Ad hoc*

The Architecture Compliance review is typically targeted for a point in time when business requirements and the Enterprise Architecture are reasonably firm, and the project architecture is taking shape, well before its completion.

The aim is to hold the review as soon as practical, at a stage when there is still time to correct any major errors or shortcomings, with the obvious proviso that there needs to have been some significant development of the project architecture in order for there to be something to review.

Inputs to the Architecture Compliance review may come from other parts of the standard project lifecycle, which may have an impact on timing.

42.3.3 Governance and Personnel Scenarios

In terms of the governance and conduct of the Architecture Compliance review, and the personnel involved, there are various possible scenarios:

- For smaller-scale projects, the review process could simply take the form of a series of questions that the project architects or project leaders pose to themselves, using the checklists provided below, perhaps collating the answers into some form of project report to management

  The need to conduct such a process is normally included in overall enterprise-wide IT governance policies.

- Where the project under review has not involved a practicing or full-time architect to date (for example, in an application-level project), the purpose of the review is typically to bring to bear the architectural expertise of an Enterprise Architecture function

  In such a case, the Enterprise Architecture function would be organizing, leading, and conducting the review, with the involvement of business domain experts. In such a scenario, the review is not a substitute for the involvement of architects in a project, but it can be a supplement or a guide to their involvement. It is probable that a database will be necessary to manage the volume of data that would be produced in the analysis of a large system or set of systems.

- In most cases, particularly in larger-scale projects, the architecture function will have been deeply involved in, and perhaps leading, the development project under review

  (This is the typical TOGAF scenario.) In such cases, the review will be co-ordinated by the lead Enterprise Architect, who will assemble a team of business and technical domain experts for the review, and compile the answers to the questions posed during the review into some form of report. The questions will typically be posed during the review by the business and technical domain experts. Alternatively, the review might be led by a representative of an Architecture Board or some similar body with enterprise-wide responsibilities.

In all cases, the Architecture Compliance review process needs the backing of senior management, and will typically be mandated as part of corporate Architecture Governance policies (see Chapter 44). Normally, the enterprise CIO or Enterprise Architecture Board (see Chapter 41) will mandate architecture reviews for all major projects, with subsequent annual reviews.

42.4 Architecture Compliance Review Process

42.4.1 Overview

The Architecture Compliance review process is illustrated in Figure 42-2.

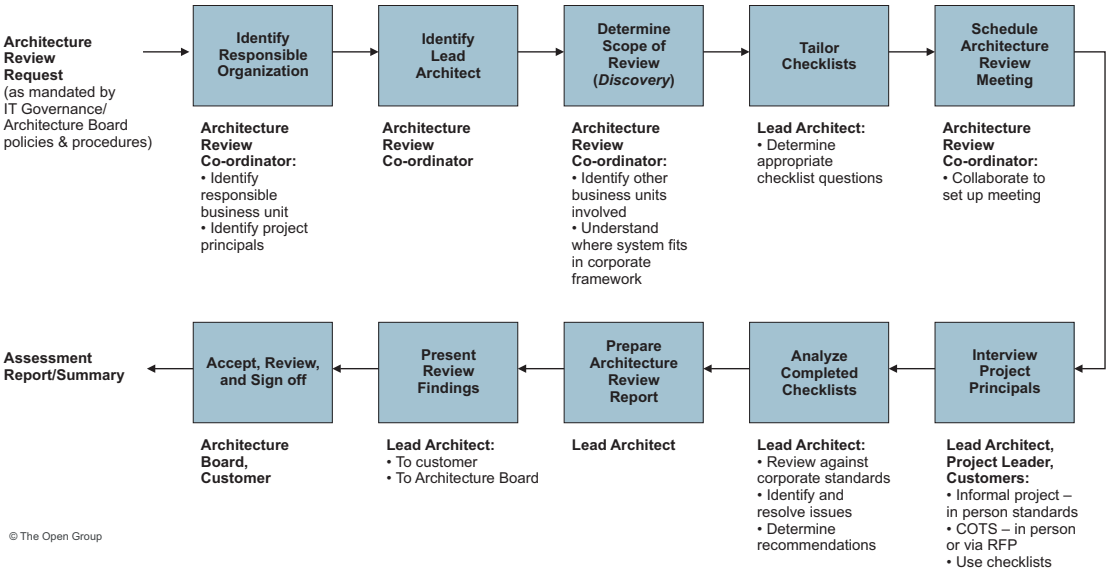

Figure 42-2 Architecture Compliance Review Process

42.4.2 Roles

The main roles in the process are tabulated below.

No.	Role	Responsibilities	Notes
1	Architecture Board	To ensure that IT architectures are consistent and support overall business needs.	Sponsor and monitor architecture activities.
2	Project Leader (or Project Board)	Responsible for the whole project.	
3	Architecture Review Co-ordinator	To administer the whole architecture development and review process.	More likely to be business-oriented than technology-oriented.
4	Lead Enterprise Architect	To ensure that the architecture is technically coherent and future-proof.	An IT architecture specialist.
5	Architect	One of the Lead Enterprise Architect's technical assistants.	
6	Customer	To ensure that business requirements are clearly expressed and understood.	Manages that part of the organization that will depend on the success of the IT described in the architecture.
7	Business Domain Expert	To ensure that the processes to satisfy the business requirements are justified and understood.	Knows how the business domain operates; may also be the customer.
8	Project Principals	To ensure that the architects have a sufficiently detailed understanding of the customer department's processes. They can provide input to the business domain expert or to the architects.	Members of the customer's organization who have input to the business requirements that the architecture is to address.

42.4.3 Steps

The main steps in the process are tabulated below.

No.	Action	Notes	Who
1	Request architecture review.	As mandated by IT governance policies and procedures.	Anyone, whether IT or business-oriented, with an interest in or responsibility for the business area affected
2	Identify responsible part of organization and relevant project principals.		Architecture Review Co-ordinator
3	Identify Lead Enterprise Architect and other architects.		Architecture Review Co-ordinator
4	Determine scope of review.	Identify which other business units/departments are involved. Understand where the system fits in the corporate architecture framework.	Architecture Review Co-ordinator
5	Tailor checklists.	To address the business requirements.	Lead Enterprise Architect
6	Schedule Architecture Review Meeting.		Architecture Review Co-ordinator with collaboration of Lead Enterprise Architect
7	Interview project principals.	To get background and technical information: ■ For internal project: in person ■ For COTS: in person or via RFP Use checklists.	Lead Enterprise Architect and/or Architect, Project Leader, and Customers
8	Analyze completed checklists.	Review against corporate standards. Identify and resolve issues. Determine recommendations.	Lead Enterprise Architect
9	Prepare Architecture Compliance review report.	May involve supporting staff.	Lead Enterprise Architect
10	Present review findings.	■ To Customer ■ To Architecture Board	Lead Enterprise Architect
11	Accept review and sign off.		Architecture Board and Customer
12	Send assessment report/summary to Architecture Review Co-ordinator.		Lead Enterprise Architect

42.5 Architecture Compliance Review Checklists

The following review checklists provide a wide range of typical questions that may be used in conducting Architecture Compliance reviews, relating to various aspects of the architecture. The organization of the questions includes the basic disciplines of system engineering, information management, security, and systems management. The checklists are based on material provided by a member of The Open Group, and are specific to that organization. Other organizations could use the following checklists with other questions tailored to their own particular needs.

The checklists provided contain too many questions for any single review: they are intended to be tailored selectively to the project concerned (see Section 42.6). The checklists actually used will typically be developed/selected by subject matter experts. They are intended to be updated annually by interest groups in those areas.

Some of the checklists include a brief description of the Architecture Principle that provokes the question, and a brief description of what to look for in the answer. These extensions to the checklist are intended to allow the intelligent re-phrasing of the questions, and to give the user of the checklist a feel for why the question is being asked.

Occasionally the questions will be written, as in RFPs, or in working with a senior project architect. More typically they are expressed orally, as part of an interview or working session with the project.

The checklists provided here are designed for use in individual architecture projects, not for business domain architecture or for architecture across multiple projects. (Doing an architecture review for a larger sphere of activity, across multiple business processes and system projects, would involve a similar process, but the checklist categories and their contents would be different.)

42.5.1 Hardware and Operating System Checklist

1. What is the project's lifecycle approach?
2. At what stage is the project in its lifecycle?
3. What key issues have been identified or analyzed that the project believes will drive evaluations of hardware and operating systems for networks, servers, and end-user devices?
4. What system capabilities will involve high-volume and/or high-frequency data transfers?
5. How does the system design impact or involve end-user devices?
6. What is the quantity and distribution (regional and global) of usage, data storage, and processing?
7. What applications are affinitized with your project by similarities in data, application services, etc.? To what degree is data affinitized with your project?
8. What hardware and operating system choices have been made before functional design of key elements of the system?
9. If hardware and operating system decisions were made outside of the project's control:
   - What awareness does the project have of the rationale for those decisions?
   - How can the project influence those decisions as system design takes shape?

10. If some non-standards have been chosen:
    - What are the essential business and technical requirements for not using corporate standards?
    - Is this supported by a business case?
    - Have the assumptions in the business case been subject to scrutiny?
11. What is your process for evaluating full lifecycle costs of hardware and operating systems?
12. How has corporate financial management been engaged in evaluation of lifecycle costs?
13. Have you performed a financial analysis of the supplier?
14. Have you made commitments to any supplier?
15. Do you believe your requirements can be met by only one supplier?

42.5.2 Software Services and Middleware Checklist

1. Describe how error conditions are defined, raised, and propagated between application components.
2. Describe the general pattern of how methods are defined and arranged in various application modules.
3. Describe the general pattern for how method parameters are defined and organized in various application modules. Are [in], [in/out], [out] parameters always specified in the same order? Do Boolean values returned by modules have a consistent outcome?
4. Describe the approach that is used to minimize the number of round-trips between client and server calls, particularly for out-of-process calls, and when complex data structures are involved.
5. Describe the major data structures that are passed between major system components.
6. Describe the major communication protocols that are used between major system components.
7. Describe the marshaling techniques that are used between various system components. Describe any specialized marshaling arrangements that are used.
8. Describe to what extent the system is designed with stateful and stateless components.
9. Describe how and when state is saved for both stateful and stateless components.
10. Describe the extent to which objects are created, used, and destroyed *versus* re-used through object pooling.
11. Describe the extent to which the system relies on threading or critical section coding.
12. Describe the approach and the internal documentation that is used internally in the system to document the methods, methods arguments, and method functionality.
13. Describe the code review process that was used to build the system.
14. Describe the unit testing that has been used to test the system components.
15. Describe the pre- and post-condition testing that is included in various system modules.

16. Describe the assertion testing that is included with the system.
17. Do components support all the interface types they need to support or are certain assumptions made about what types of components will call other components either in terms of language bindings or other forms of marshaling?
18. Describe the extent to which big-endian or little-endian data format problems need to be handled across different platforms.
19. Describe if numbers or strings need to be handled differently across different platforms.
20. Describe whether the software needs to check for floating-point round-off errors.
21. Describe how time and date functions manage dates so as to avoid improper handling of time and date calculation or display.
22. Describe what tools or processes have been used to test the system for memory leaks, reachability, or general robustness.
23. Describe the layering of the systems services software. Describe the general number of links between major system components. Is the system composed of a lot of point-to-point interfaces or are major messaging backbones used instead?
24. Describe to what extent the system components are either loosely coupled or tightly coupled.
25. What requirements does the system need from the infrastructure in terms of shared libraries, support for communication protocols, load balancing, transaction processing, system monitoring, naming services, or other infrastructure services?
26. Describe how the system and system components are designed for refactoring.
27. Describe how the system or system components rely on common messaging infrastructure *versus* a unique point-to-point communication structure.

42.5.3 Applications Checklists

42.5.3.1 Infrastructure (Enterprise Productivity) Applications

1. Is there need for capabilities that are not provided through the enterprise's standard infrastructure application products? For example:
   - Collaboration
     - Application sharing
     - Video conferencing
     - Calendaring
     - Email
   - Workflow management
   - Publishing/word processing applications
     - HTML
     - SGML and XML
     - Portable document format

- — Document processing (proprietary format)
- — Desktop publishing

- Spreadsheet applications
- Presentation applications
  - — Business presentations
  - — Image
  - — Animation
  - — Video
  - — Sound
  - — CBT
  - — Web browsers
- Data management applications
  - — Database interface
  - — Document management
  - — Product data management
  - — Data warehouses/mart
- Program management applications
  - — Project management
  - — Program visibility

2. Describe the business requirements for enterprise infrastructure application capabilities that are not met by the standard products.

42.5.3.2 Business Applications

1. Are any of the capabilities required provided by standard products supporting one or more line-of-business applications? For example:
   - Business acquisition applications
     - — Sales and marketing
   - Engineering applications
     - — Computer-aided design
     - — Computer-aided engineering
     - — Mathematical and statistics analysis
   - Supplier management applications
     - — Supply chain management
     - — Customer relationship management
   - Manufacturing applications

- — Enterprise Resource Planning (ERP) applications
- — Manufacturing execution systems
- — Manufacturing quality
- — Manufacturing process engineering
- — Machine and adaptive control

- Customer support applications
  - — Airline logistics support
  - — Maintenance engineering
- Finance applications
- People applications
- Facilities applications
- Information systems applications
  - — Systems engineering
  - — Software engineering
  - — Web developer tools
  - — Integrated development environments
  - — Lifecycle categories
  - — Functional categories
  - — Specialty categories
- Computer-aided manufacturing
- e-Business enablement
- Business process engineering
  - — Statistical quality control

2. Describe the process requirements for business application capabilities that are not met by the standard products.

42.5.3.3 Application Integration Approach

1. What integration points (business process/activity, application, data, computing environment) are targeted by this architecture?
2. What application integration techniques will be applied (common business objects [ORBs], standard data definitions [XML, etc.], common user interface presentation/desktop)?

42.5.4 Information Management Checklists

42.5.4.1 Data Values

1. What are the processes that standardize the management and use of the data?
2. What business process supports the entry and validation of the data? Use of the data?
3. What business actions correspond to the creation and modification of the data?
4. What business actions correspond to the deletion of the data and is it considered part of a business record?
5. What are the data quality requirements required by the business user?
6. What processes are in place to support data referential integrity and/or normalization?

42.5.4.2 Data Definition

1. What are the data model, data definitions, structure, and hosting options of purchased applications (COTS)?
2. What are the rules for defining and maintaining the data requirements and designs for all components of the information system?
3. What shareable repository is used to capture the model content and the supporting information for data?
4. What is the physical data model definition (derived from logical data models) used to design the database?
5. What software development and data management tools have been selected?
6. What data owners have been identified to be responsible for common data definitions, eliminating unplanned redundancy, providing consistently reliable, timely, and accurate information, and protecting data from misuse and destruction?

42.5.4.3 Security/Protection

1. What are the data entity and attribute access rules which protect the data from unintentional and unauthorized alterations, disclosure, and distribution?
2. What are the data protection mechanisms to protect data from unauthorized external access?
3. What are the data protection mechanisms to control access to data from external sources that temporarily have internal residence within the enterprise?

42.5.4.4 Hosting, Data Types, and Sharing

1. What is the discipline for managing sole-authority data as one logical source with defined updating rules for physical data residing on different platforms?
2. What is the discipline for managing replicated data, which is derived from operational sole-authority data?

3. What tier data server has been identified for the storage of high or medium-critical operational data?
4. What tier data server has been identified for the storage of type C operational data?
5. What tier data server has been identified for the storage of decision support data contained in a data warehouse?
6. What Database Management Systems (DBMSs) have been implemented?

42.5.4.5 Common Services

1. What are the standardized distributed data management services (e.g., validation, consistency checks, data edits, encryption, and transaction management) and where do they reside?

42.5.4.6 Access Method

1. What are the data access requirements for standard file, message, and data management?
2. What are the access requirements for decision support data?
3. What are the data storage and the application logic locations?
4. What query language is being used?

42.5.5 Security Checklist

1. **Security Awareness**: Have you ensured that the corporate security policies and guidelines to which you are designing are the latest versions? Have you read them? Are you aware of all relevant computing security compliance and risk acceptance processes? (Interviewer should list all relevant policies and guidelines.)
2. **Identification/Authentication**: Diagram the process flow of how a user is identified to the application and how the application authenticates that the user is who they claim to be. Provide supporting documentation to the diagram explaining the flow from the user interface to the application/database server(s) and back to the user. Are you compliant with corporate policies on accounts, passwords, etc.?
3. **Authorization**: Provide a process flow from beginning to end showing how a user requests access to the application, indicating the associated security controls and separation of duties. This should include how the request is approved by the appropriate data owner, how the user is placed into the appropriate access-level classification profile, how the user ID, password, and access is created and provided to the user. Also include how the user is informed of their responsibilities associated with using the application, given a copy of the access agreement, how to change password, who to call for help, etc.
4. **Access Controls**: Document how the user IDs, passwords, and access profiles are added, changed, removed, and documented. The documentation should include who is responsible for these processes.
5. **Sensitive Information Protection**: Provide documentation that identifies sensitive data requiring additional protection. Identify the data owners responsible for this data and the process to be used to protect storage, transmission, printing, and distribution of this data. Include how the password file/field is protected. How will users be prevented from viewing someone else's sensitive information? Are there agreements with outside parties

(partners, suppliers, contractors, etc.) concerning the safeguarding of information? If so, what are the obligations?

6. **Audit Trails and Audit Logs**: Identify and document group accounts required by the users or application support, including operating system group accounts. Identify and document individual accounts and/or roles that have superuser type privileges, what these privileges are, who has access to these accounts, how access to these accounts is controlled, tracked, and logged, and how password change and distribution are handled, including operating system accounts. Also identify audit logs, who can read the audit logs, who can modify the audit logs, who can delete the audit logs, and how the audit logs are protected and stored. Is the user ID obscured in the audit trails?
7. **External Access Considerations**: Will the application be used internally only? If not, are you compliant with corporate external access requirements?

42.5.6 System Management Checklist

1. What is the frequency of software changes that must be distributed?
2. What tools are used for software distribution?
3. Are multiple software and/or data versions allowed in production?
4. What is the user data backup frequency and expected restore time?
5. How are user accounts created and managed?
6. What is the system license management strategy?
7. What general system administration tools are required?
8. What specific application administration tools are required?
9. What specific service administration tools are required?
10. How are service calls received and dispatched?
11. Describe how the system is uninstalled.
12. Describe the process or tools available for checking that the system is properly installed.
13. Describe tools or instrumentation that are available that monitor the health and performance of the system.
14. Describe the tools or process in place that can be used to determine where the system has been installed.
15. Describe what form of audit logs are in place to capture system history, particularly after a mishap.
16. Describe the capabilities of the system to dispatch its own error messages to service personnel.

42.5.7 System Engineering/Overall Architecture Checklists

42.5.7.1 General

1. What other applications and/or systems require integration with yours?
2. Describe the integration level and strategy with each.
3. How geographically distributed is the user base?
4. What is the strategic importance of this system to other user communities inside or outside the enterprise?
5. What computing resources are needed to provide system service to users inside the enterprise? Outside the enterprise and using enterprise computing assets? Outside the enterprise and using their own assets?
6. How can users outside the native delivery environment access your applications and data?
7. What is the life expectancy of this application?
8. Describe the design that accommodates changes in the user base, stored data, and delivery system technology.
9. What is the size of the user base and their expected performance level?
10. What performance and stress test techniques do you use?
11. What is the overall organization of the software and data components?
12. What is the overall service and system configuration?
13. How are software and data configured and mapped to the service and system configuration?
14. What proprietary technology (hardware and software) is needed for this system?
15. Describe how each and every version of the software can be reproduced and re-deployed over time.
16. Describe the current user base and how that base is expected to change over the next three to five years.
17. Describe the current geographic distribution of the user base and how that base is expected to change over the next three to five years.
18. Describe how many current or future users need to use the application in a mobile capacity or who need to work off-line.
19. Describe what the application generally does, the major components of the application, and the major data flows.
20. Describe the instrumentation included in the application that allows for the health and performance of the application to be monitored.
21. Describe the business justification for the system.
22. Describe the rationale for picking the system development language over other options in terms of initial development cost *versus* long-term maintenance cost.
23. Describe the systems analysis process that was used to come up with the system architecture and product selection phase of the system architecture.

24. Who besides the original customer might have a use for or benefit from using this system?
25. What percentage of the users use the system in browse mode *versus* update mode?
26. What is the typical length of requests that are transactional?
27. Do you need guaranteed data delivery or update, or does the system tolerate failure?
28. What are the up-time requirements of the system?
29. Describe where the system architecture adheres or does not adhere to standards.
30. Describe the project planning and analysis approach used on the project.

42.5.7.2 Processors/Servers/Clients

1. Describe the client/server Application Architecture.
2. Annotate the pictorial to illustrate where application functionality is executed.

42.5.7.3 Client

1. Are functions other than presentation performed on the user device?
2. Describe the data and process help facility being provided.
3. Describe the screen-to-screen navigation technique.
4. Describe how the user navigates between this and other applications.
5. How is this and other applications launched from the user device?
6. Are there any inter-application data and process sharing capabilities? If so, describe what is being shared and by what technique/technology.
7. Describe data volumes being transferred to the client.
8. What are the additional requirements for local data storage to support the application?
9. What are the additional requirements for local software storage/memory to support the application?
10. Are there any known hardware/software conflicts or capacity limitations caused by other application requirements or situations which would affect the application users?
11. Describe how the look-and-feel of your presentation layer compares to the look-and-feel of the other existing applications.
12. Describe to what extent the client needs to support asynchronous and/or synchronous communication.
13. Describe how the presentation layer of the system is separated from other computational or data transfer layers of the system.

42.5.7.4 Application Server

1. Can/do the presentation layer and application layers run on separate processors?
2. Can/do the application layer and data access layer run on separate processors?
3. Can this application be placed on an application server independent of all other applications? If not, explain the dependencies.
4. Can additional parallel application servers be easily added? If so, what is the load balancing mechanism?
5. Has the resource demand generated by the application been measured and what is the value? If so, has the capacity of the planned server been confirmed at the application and aggregate levels?

42.5.7.5 Data Server

1. Are there other applications which must share the data server? If so, identify them and describe the data and data access requirements.
2. Has the resource demand generated by the application been measured and what is the value? If so, has the capacity of the planned server been confirmed at the application and aggregate levels?

42.5.7.6 COTS (where applicable)

1. Is the vendor substantial and stable?
2. Will the enterprise receive source code upon demise of the vendor?
3. Is this software configured for the enterprise's usage?
4. Is there any peculiar A&D data or processes that would impede the use of this software?
   — Is this software currently available?
5. Has it been used/demonstrated for volume/availability/service-level requirements similar to those of the enterprise?
   — Describe the past financial and market share history of the vendor.

42.5.8 System Engineering/Methods & Tools Checklist

1. Do metrics exist for the current way of doing business?
2. Has the system owner created evaluation criteria that will be used to guide the project? Describe how the evaluation criteria will be used.
3. Has research of existing architectures been done to leverage existing work? Describe the method used to discover and understand. Will the architectures be integrated? If so, explain the method that will be used.
4. Describe the methods that will be used on the project:
   — For defining business strategies

- For defining areas in need of improvement
- For defining baseline and target business processes
- For defining transition processes
- For managing the project
- For team communication
- For knowledge management, change management, and configuration management
- For software development
- For referencing standards and statements of direction
- For quality assurance of deliverables
- For design reviews and deliverable acceptance
- For capturing metrics

5. Are the methods documented and distributed to each team member?
6. To what extent are team members familiar with these methods?
7. What processes are in place to ensure compliance with the methods?
8. Describe the infrastructure that is in place to support the use of the methods through the end of the project and anticipated releases.
   - How is consultation and trouble-shooting provided?
   - How is training co-ordinated?
   - How are changes and enhancements incorporated and cascaded?
   - How are lessons learned captured and communicated?
9. What tools are being used on the project? (Specify versions and platforms). To what extent are team members familiar with these tools?
10. Describe the infrastructure that is in place to support the use of the tools through the end of the project and anticipated releases.
    - How is consultation and trouble-shooting provided?
    - How is training co-ordinated?
    - How are changes and enhancements incorporated and cascaded?
    - How are lessons learned captured and communicated?
11. Describe how the project will promote the re-use of its deliverables and deliverable content.
12. Will the architecture designs "live" after the project has been implemented? Describe the method that will be used to incorporate changes back into the architecture designs.
13. Were the current processes defined?
14. Were issues documented, rated, and associated to current processes? If not, how do you know you are fixing something that is broken?
15. Were existing/planned process improvement activities identified and associated to current processes? If not, how do you know this activity is not in conflict with or redundant to other Statements of Work?

16. Do you have current metrics? Do you have forecasted metrics? If not, how do you know you are improving something?
17. What processes will you put in place to gather, evaluate, and report metrics?
18. What impacts will the new design have on existing business processes, organizations, and information systems? Have they been documented and shared with the owners?

42.6 Architecture Compliance Review Guidelines

42.6.1 Tailoring the Checklists

- Focus on:
  - High risk areas
  - Expected (and emergent) differentiators
- For each question in the checklist, understand:
  - The question itself
  - The principle behind it
  - What to look for in the responses
- Ask subject experts for their views
- Fix the checklist questions for your use
- Bear in mind the need for feedback to the Architecture Board

42.6.2 Conducting Architecture Compliance Reviews

- Understand clearly the objectives of those soliciting the review; and stay on track and deliver what was asked for. For example, they typically want to know what is right or wrong with the system being architected; not what is right or wrong with the development methodology used, their own management structure, etc. It is easy to get off-track and discuss subjects that are interesting and perhaps worthwhile, but not what was solicited. If you can shed light and insight on technical approaches, but the discussion is not necessary for the review, volunteer to provide it after the review.
- If it becomes obvious during the discussion that there are other issues that need to be addressed, which are outside the scope of the requested review, bring it up with the meeting chair afterwards. A plan for addressing the issues can then be developed in accordance with their degree of seriousness.
- Stay "scientific". Rather than: "We like to see large databases hosted on *ABC* rather than *XYZ*.", say things like: "The downtime associated with *XYZ* database environments is much greater than on *ABC* database environments. Therefore we don't recommend hosting type *M* and *N* systems in an *XYZ* environment."
- Ask "open" questions; i.e., questions that do not presume a particular answer.
- There are often "hidden agendas" or controversial issues among those soliciting a review, which you probably won't know up-front. A depersonalized approach to the discussions may help bridge the gaps of opinion rather than exacerbate them.

- Treat those being interviewed with respect. They may not have built the system "the way it should be", but they probably did the best they could under the circumstances in which they were placed.
- Help the exercise become a learning experience for you and the presenters.
- Reviews should include detailed assessment activities against the architectures and should ensure that the results are stored in the Enterprise Continuum.

Chapter 43

Architecture Contracts

This chapter provides guidelines for defining and using Architecture Contracts.

43.1 Role

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 Chapter 44). 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

The traditional Architecture Contract is an agreement between the sponsor and the architecture function or IS department. However, increasingly more services are being provided by systems integrators, applications providers, and service providers, co-ordinated through the architecture function or IS department. There is therefore a need for an Architecture Contract to establish joint agreements between all parties involved in the architecture development and delivery.

Architecture Contracts may occur at various stages of the Architecture Development Method (ADM); for example:

- The Statement of Architecture Work created in Phase A of Part II: Architecture Development Method (ADM) is effectively an Architecture Contract between the architecting organization and the sponsor of the Enterprise Architecture (or the IT governance function)
- The development of one or more architecture domains (business, data, application, technology), and in some cases the oversight of the overall Enterprise Architecture, may be contracted out to systems integrators, applications providers, and/or service providers

  Each of these arrangements will normally be governed by an Architecture Contract that

defines the deliverables, quality, and fitness-for-purpose of the developed architecture, and the processes by which the partners in the architecture development will work together.

- At the beginning of Phase G (Implementation Governance), between the architecture function and the function responsible for implementing the Enterprise Architecture defined in the preceding ADM phases; typically, this will be either the in-house systems development function, or a major contractor to whom the work is outsourced
  - What is being "implemented" in Phase G of the ADM is the overall Enterprise Architecture

    This will typically include the technology infrastructure (from Phase D), and also those enterprise applications and data management capabilities that have been defined in the Application Architecture and Data Architecture (from Phase C), either because they are enterprise-wide in scope, or because they are strategic in business terms, and therefore of enterprise-wide importance and visibility. However, it will typically not include non-strategic business applications, which business units will subsequently deploy on top of the technology infrastructure that is implemented as part of the Enterprise Architecture.
  - In larger-scale implementations, there may well be one Architecture Contract per implementation team in a program of implementation projects
- When the Enterprise Architecture has been implemented (at the end of Phase G), an Architecture Contract will normally be drawn up between the architecting function (or the IT governance function, subsuming the architecting function) and the business users who will subsequently be building and deploying application systems in the architected environment

It is important to bear in mind in all these cases that the ultimate goal is not just an Enterprise Architecture, but a dynamic Enterprise Architecture; i.e., one that allows for flexible evolution in response to changing technology and business drivers, without unnecessary constraints. The Architecture Contract is crucial to enabling a dynamic Enterprise Architecture and is key to governing the implementation.

Typical contents of these three kinds of Architecture Contract are explained below.

43.2 Contents

43.2.1 Statement of Architecture Work

The Statement of Architecture Work is created as a deliverable of Phase A, and is effectively an Architecture Contract between the architecting organization and the sponsor of the Enterprise Architecture (or the IT governance function, on behalf of the enterprise).

The typical contents of a Statement of Architecture Work are as defined in Part IV, Section 32.2.20.

43.2.2 Contract between Architecture Design and Development Partners

This is a signed statement of intent on designing and developing the Enterprise Architecture, or significant parts of it, from partner organizations, including systems integrators, applications providers, and service providers.

Increasingly, the development of one or more architecture domains (business, data, application, technology) may be contracted out, with the enterprise's architecture function providing oversight of the overall Enterprise Architecture, and co-ordination and control of the overall effort. In some cases even this oversight role may be contracted out, although most enterprises prefer to retain that core responsibility in-house.

Whatever the specifics of the contracting-out arrangements, the arrangements themselves will normally be governed by an Architecture Contract that defines the deliverables, quality, and fitness-for-purpose of the developed architecture, and the processes by which the partners in the architecture development will work together.

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

The template for this contract will normally be defined as part of the Preliminary Phase of the ADM, if not existing already, and the specific contract will be defined at the appropriate stage of the ADM, depending on the particular work that is being contracted out.

43.2.3 Contract between Architecting Function and Business Users

This is a signed statement of intent to conform with the Enterprise Architecture, issued by enterprise business users. When the Enterprise Architecture has been implemented (at the end of Phase F), an Architecture Contract will normally be drawn up between the architecting function (or the IT governance function, subsuming the architecting function) and the business users who will subsequently be building and deploying application systems in the architected environment.

Typical contents of a Business Users' Architecture Contract are:

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

This contract is also used to manage changes to the Enterprise Architecture in Phase H.

43.3 Relationship to Architecture Governance

The Architecture Contract document produced in Phase G of the ADM figures prominently in the area of Architecture Governance, as explained in Part VI, Chapter 44.

In the context of Architecture Governance, the Architecture Contract is often used as a means of driving architecture change.

In order to ensure that the Architecture Contract is effective and efficient, the following aspects of the governance framework may need to be introduced into Phase G:

- Simple processes
- People-centered authority
- Strong communication
- Timely responses and an effective escalation process
- Supporting organizational structures
- Status tracking of architecture implementation

Chapter 44

Architecture Governance

This chapter provides a framework and guidelines for Architecture Governance.

44.1 Introduction

This section describes the nature of governance, and the levels of governance.

44.1.1 Levels of Governance within the Enterprise

Architecture Governance is the practice and orientation by which Enterprise Architectures and other architectures are managed and controlled at an enterprise-wide level.

Architecture Governance typically does not operate in isolation, but within a hierarchy of governance structures, which, particularly in the larger enterprise, can include all of the following as distinct domains with their own disciplines and processes:

- Corporate Governance
- Technology Governance
- IT Governance
- Architecture Governance

Each of these domains of governance may exist at multiple geographic levels — global, regional, and local — within the overall enterprise.

Corporate governance is thus a broad topic, beyond the scope of an Enterprise Architecture framework such as the TOGAF framework.

This and related subsections are focused on Architecture Governance; but they describe it in the context of enterprise-wide governance, because of the hierarchy of governance structures within which it typically operates, as explained above.

In particular, this and following sections aim to:

- Provide an overview of the nature of governance as a discipline in its own right
- Describe the governance context in which Architecture Governance typically functions within the enterprise
- Describe an Architecture Governance Framework that can be adapted and applied in practice, both for Enterprise Architecture and for other forms of IT architecture

44.1.2 Nature of Governance

44.1.2.1 Governance: A Generic Perspective

Governance is essentially about ensuring that business is conducted properly. It is less about overt control and strict adherence to rules, and more about guidance and effective and equitable usage of resources to ensure sustainability of an organization's strategic objectives.

The following outlines the basic principles of corporate governance, as identified by the Organization for Economic Co-operation and Development (OECD):

- Focuses on the rights, roles, and equitable treatment of shareholders
- Disclosure and transparency and the responsibilities of the board
- Ensures:
  - Sound strategic guidance of the organization
  - Effective monitoring of management by the board
  - Board accountability for the company and to the shareholders
- Board's responsibilities:
  - Reviewing and guiding corporate strategy
  - Setting and monitoring achievement of management's performance objectives

Supporting this, the OECD considers a traditional view of governance as: "... the system by which business corporations are directed and controlled. The corporate governance structure specifies the distribution of rights and responsibilities among different participants in the corporation — such as the board, managers, shareholders, and other stakeholders — and spells out the rules and procedures for making decisions on corporate affairs. By doing this, it also provides the structure through which the company objectives are set, and the means of attaining those objectives and monitoring performance" (Source: OECD, 2001).

44.1.2.2 Characteristics of Governance

The following characteristics have been adapted from *Corporate Governance* (Naidoo, 2002) and are positioned here to highlight both the value and necessity for governance as an approach to be adopted within organizations and their dealings with all involved parties:

Discipline All involved parties will have a commitment to adhere to procedures, processes, and authority structures established by the organization.

Transparency All actions implemented and their decision support will be available for inspection by authorized organization and provider parties.

Independence All processes, decision-making, and mechanisms used will be established so as to minimize or avoid potential conflicts of interest.

Accountability Identifiable groups within the organization — e.g., governance boards who take actions or make decisions — are authorized and accountable for their actions.

Responsibility Each contracted party is required to act responsibly to the organization and its stakeholders.

Fairness All decisions taken, processes used, and their implementation will not be allowed to create unfair advantage to any one particular party.

44.1.3 Technology Governance

Technology governance controls how an organization utilizes technology in the research, development, and production of its goods and services. Although it may include IT governance activities, it often has broader scope.

Technology governance is a key capability, requirement, and resource for most organizations because of the pervasiveness of technology across the organizational spectrum.

Recent studies have shown that many organizations have a balance in favor of intangibles rather than tangibles that require management. Given that most of these intangibles are informational and digital assets, it is evident that businesses are becoming more reliant on IT: and the governance of IT — IT governance — is therefore becoming an even more important part of technology governance.

These trends also highlight the dependencies of businesses on not only the information itself but also the processes, systems, and structures that create, deliver, and consume it. As the shift to increasing value through intangibles increases in many industry sectors, so risk management must be considered as key to understanding and moderating new challenges, threats, and opportunities.

Not only are organizations increasingly dependent on IT for their operations and profitability, but also their reputation, brand, and ultimately their values are also dependent on that same information and the supporting technology.

44.1.4 IT Governance

IT governance provides the framework and structure that links IT resources and information to enterprise goals and strategies. Furthermore, IT governance institutionalizes best practices for planning, acquiring, implementing, and monitoring IT performance, to ensure that the enterprise's IT assets support its business objectives.

In recent years, IT governance has become integral to the effective governance of the modern enterprise. Businesses are increasingly dependent on IT to support critical business functions and processes; and to successfully gain competitive advantage, businesses need to manage effectively the complex technology that is pervasive throughout the organization, in order to respond quickly and safely to business needs.

In addition, regulatory environments around the world are increasingly mandating stricter enterprise control over information, driven by increasing reports of information system disasters and electronic fraud. The management of IT-related risk is now widely accepted as a key part of enterprise governance.

It follows that an IT governance strategy, and an appropriate organization for implementing the strategy, must be established with the backing of top management, clarifying who owns the enterprise's IT resources, and, in particular, who has ultimate responsibility for their enterprise-wide integration.

44.1.4.1 An IT Controls Framework — COBIT

As with corporate governance, IT governance is a broad topic, beyond the scope of an Enterprise Architecture framework such as the TOGAF framework. A good source of detailed information on IT governance is the COBIT framework (Control OBjectives for Information and related Technology). This is an open standard for control over IT, developed and promoted by the IT Governance Institute (ITGI), and published by the Information Systems Audit and Control Foundation (ISACF). COBIT controls may provide useful aides to running a compliance strategy.

44.1.5 Architecture Governance: Overview

44.1.5.1 Architecture Governance Characteristics

Architecture Governance is the practice and orientation by which Enterprise Architectures and other architectures are managed and controlled at an enterprise-wide level. It includes the following:

- Implementing a system of controls over the creation and monitoring of all architectural components and activities, to ensure the effective introduction, implementation, and evolution of architectures within the organization
- Implementing a system to ensure compliance with internal and external standards and regulatory obligations
- Establishing processes that support effective management of the above processes within agreed parameters
- Developing practices that ensure accountability to a clearly identified stakeholder community, both inside and outside the organization

44.1.5.2 Architecture Governance as a Board-Level Responsibility

As mentioned above, IT governance has recently become a board responsibility as part of overall business governance. The governance of an organization's architectures is a key factor in effective IT/business linkage, and is therefore increasingly becoming a key board-level responsibility in its own right.

This section aims to provide the impetus for opening up IT and Architecture Governance so that the business responsibilities associated with architecture activities and artifacts can be elucidated and managed.

44.1.5.3 The TOGAF Standard and Architecture Governance

Phase G of the TOGAF ADM (see Part II, Chapter 14) is dedicated to implementation governance, which concerns itself with the realization of the architecture through change projects. Implementation governance is just one aspect of Architecture Governance, which covers the management and control of all aspects of the development and evolution of Enterprise Architectures and other architectures within the enterprise.

Architecture Governance needs to be supported by an Architecture Governance Framework (described in Section 44.2) which assists in identifying effective processes so that the business responsibilities associated with Architecture Governance can be elucidated, communicated, and managed effectively.

44.2 Architecture Governance Framework

This section describes a conceptual and organizational framework for Architecture Governance

As previously explained, Phase G of the TOGAF ADM (see Part II, Chapter 14) is dedicated to implementation governance, which concerns itself with the realization of the architecture through change projects.

Implementation governance is just one aspect of Architecture Governance, which covers the management and control of all aspects of the development and evolution of Enterprise Architectures and other architectures within the enterprise.

Architecture Governance needs to be supported by an Architecture Governance Framework, described below. The governance framework described is a generic framework that can be adapted to the existing governance environment of an enterprise. It is intended to assist in identifying effective processes and organizational structures, so that the business responsibilities associated with Architecture Governance can be elucidated, communicated, and managed effectively.

44.2.1 Architecture Governance Framework — Conceptual Structure

44.2.1.1 Key Concepts

Conceptually, Architecture Governance is an approach, a series of processes, a cultural orientation, and set of owned responsibilities that ensure the integrity and effectiveness of the organization's architectures.

The key concepts are illustrated in Figure 44-1.

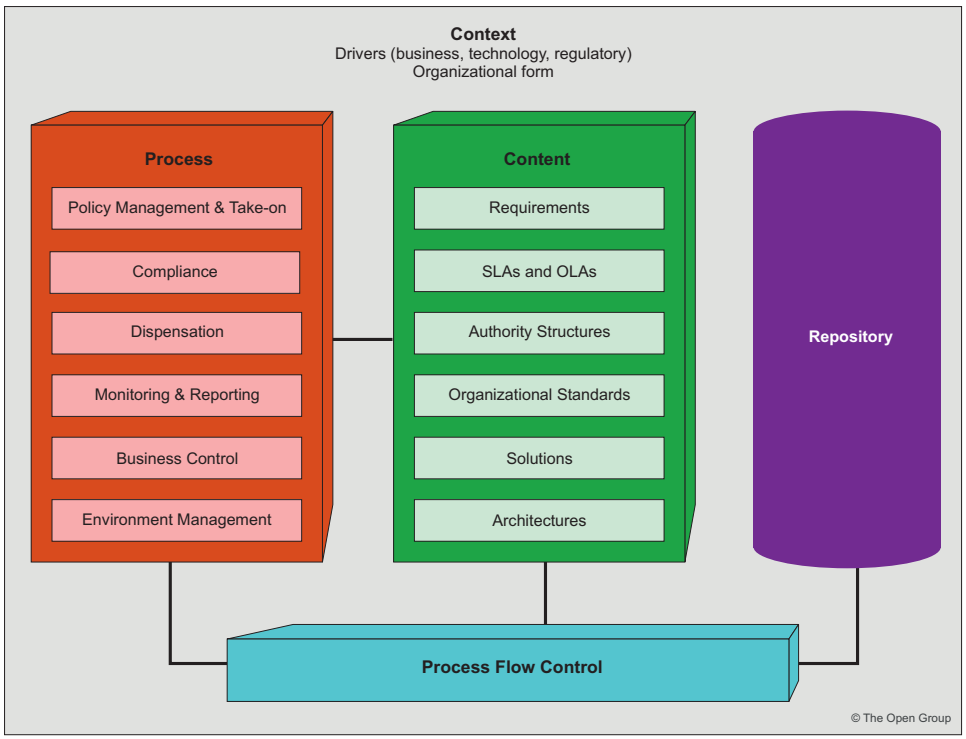

Figure 44-1 Architecture Governance Framework — Conceptual Structure

The split of process, content, and context are key to the support of the Architecture Governance initiative, by allowing the introduction of new governance material (legal, regulatory, standards-based, or legislative) without unduly impacting the processes. This content-agnostic approach ensures that the framework is flexible. The processes are typically independent of the content and implement a proven best practice approach to active governance.

The Architecture Governance Framework is integral to the Enterprise Continuum, and manages all content relevant both to the architecture itself and to Architecture Governance processes.

44.2.1.2 Key Architecture Governance Processes

Governance processes are required to identify, manage, audit, and disseminate all information related to architecture management, contracts, and implementation. These governance processes will be used to ensure that all architecture artifacts and contracts, principles, and Operational-Level Agreements (OLAs) are monitored on an ongoing basis with clear auditability of all decisions made.

Policy Management and Take-On

All architecture amendments, contracts, and supporting information must come under governance through a formal process in order to register, validate, ratify, manage, and publish new or updated content. These processes will ensure the orderly integration with existing governance content such that all relevant parties, documents, contracts, and supporting information are managed and audited.

Compliance

Compliance assessments against Service-Level Agreements (SLAs), OLAs, standards, and regulatory requirements will be implemented on an ongoing basis to ensure stability, conformance, and performance monitoring. These assessments will be reviewed and either accepted or rejected depending on the criteria defined within the governance framework.

Dispensation

A Compliance Assessment can be rejected where the subject area (design, operational, service level, or technology) are not compliant. In this case the subject area can:

1. Be adjusted or realigned in order to meet the compliance requirements
2. Request a dispensation

Where a Compliance Assessment is rejected, an alternate route to meeting interim conformance is provided through dispensations. These are granted for a given time period and set of identified service and operational criteria that must be enforced during the lifespan of the dispensation. Dispensations are not granted indefinitely, but are used as a mechanism to ensure that service levels and operational levels are met while providing a level of flexibility in their implementation and timing. The time-bound nature of dispensations ensures that they are a major trigger in the compliance cycle.

Monitoring and Reporting

Performance management is required to ensure that both the operational and service elements are managed against an agreed set of criteria. This will include monitoring against SLAs and OLAs, feedback for adjustment, and reporting.

Internal management information will be considered in Environment Management.

Business Control

Business Control relates to the processes invoked to ensure compliance with the organization's business policies.

Environment Management

This identifies all the services required to ensure that the repository-based environment underpinning the governance framework is effective and efficient. This includes the physical and logical repository management, access, communication, training, and accreditation of all users.

All architecture artifacts, service agreements, contracts, and supporting information must come under governance through a formal process in order to register, validate, ratify, manage, and publish new or updated content. These processes will ensure the orderly integration with existing governance content such that all relevant parties, documents, contracts, and supporting information are managed and audited.

The governance environment will have a number of administrative processes defined in order to effect a managed service and process environment. These processes will include user management, internal SLAs (defined in order to control its own processes), and management information reporting.

44.2.2 Architecture Governance Framework — Organizational Structure

44.2.2.1 Overview

Architecture Governance is the practice and orientation by which Enterprise Architectures and other architectures are managed and controlled. In order to ensure that this control is effective within the organization, it is necessary to have the correct organizational structures established to support all governance activities.

An Architecture Governance structure for effectively implementing the approach described in this section will typically include the following levels, which may in practice involve a combination of existing IT governance processes, organizational structures, and capabilities. They will typically include the following:

- Global governance board
- Local governance board
- Design authorities
- Working parties

The architecture organization illustrated in Figure 44-2 highlights the major structural elements required for an Architecture Governance initiative. While each enterprise will have differing requirements, it is expected that the basics of the organizational design shown in Figure 44-2 will be applicable and implementable in a wide variety of organizational types.

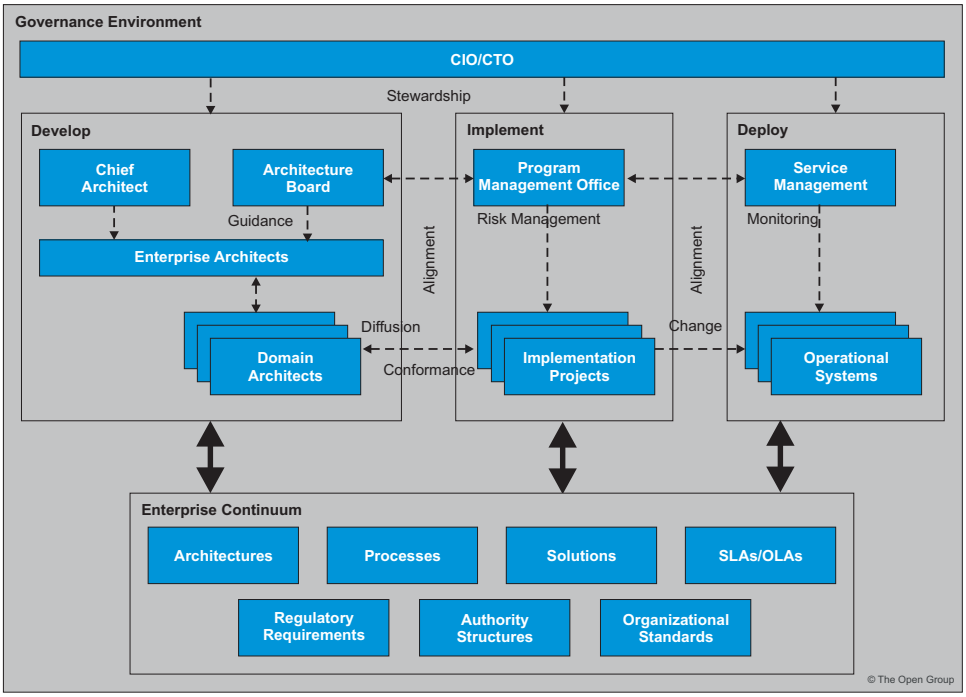

Figure 44-2 Architecture Governance Framework — Organizational Structure

44.2.2.2 Key Areas

Figure 44-2 identifies three key areas of architecture management: Develop, Implement, and Deploy. Each of these is the responsibility of one or more groups within the organization, while the Enterprise Continuum is shown to support all activities and artifacts associated with the governance of the architectures throughout their lifecycle.

The Develop responsibilities, processes, and structures are usually linked to the TOGAF ADM and its usage, while the Implement responsibilities, processes, and structures are typically linked to Phase G (see Part II, Chapter 14).

As mentioned above, the Architecture Governance Framework is integral to the Enterprise Continuum, and manages all content relevant both to the architectures themselves and to Architecture Governance processes.

44.2.2.3 Operational Benefits

As illustrated in Figure 44-2, the governance of the organization's architectures provides not only direct control and guidance of their development and implementation, but also extends into the operations of the implemented architectures.

The following benefits have been found to be derived through the continuing governance of architectures:

- Links IT processes, resources, and information to organizational strategies and objectives
- Integrates and institutionalizes IT best practices
- Aligns with industry frameworks such as COBIT (planning and organizing, acquiring and implementing, delivering and supporting, and monitoring IT performance)
- Enables the organization to take full advantage of its information, infrastructure, and hardware and software assets
- Protects the underlying digital assets of the organization
- Supports regulatory and best practice requirements such as auditability, security, responsibility, and accountability
- Promotes visible risk management

These benefits position the TOGAF Architecture Governance Framework as an approach, a series of processes, a cultural orientation, and a set of owned responsibilities, that together ensure the integrity and effectiveness of the organization's architectures.

44.3 Architecture Governance in Practice

This section provides practical guidelines for the effective implementation of Architecture Governance

44.3.1 Architecture Governance — Key Success Factors

It is important to consider the following to ensure a successful approach to Architecture Governance, and to the effective management of the Architecture Contract:

- Best practices for the submission, adoption, re-use, reporting, and retirement of architecture policies, procedures, roles, skills, organizational structures, and support services
- Organizational responsibilities and structures to support the Architecture Governance processes and reporting requirements
- Integration of tools and processes to facilitate the take-up of the processes, both procedurally and culturally
- Criteria for the control of the Architecture Governance processes, dispensations, compliance assessments, SLAs, and OLAs
- Internal and external requirements for the effectiveness, efficiency, confidentiality, integrity, availability, compliance, and reliability of all Architecture Governance-related information, services, and processes

44.3.2 Elements of an Effective Architecture Governance Strategy

44.3.2.1 Architecture Governance and Corporate Politics

An Enterprise Architecture imposed without appropriate political backing is bound to fail. In order to succeed, the Enterprise Architecture must reflect the needs of the organization. Enterprise Architects, if they are not involved in the development of business strategy, must at least have a fundamental understanding of it and of the prevailing business issues facing the organization. It may even be necessary for them to be involved in the system deployment process and to ultimately own the investment and product selection decisions arising from the implementation of the Technology Architecture.

There are three important elements of Architecture Governance strategy that relate particularly to the acceptance and success of architecture within the enterprise. While relevant and applicable in their own right apart from their role in governance, and therefore described separately, they also form an integral part of any effective Architecture Governance strategy.

- A cross-organizational Architecture Board (see Chapter 41) must be established with the backing of top management to oversee the implementation of the Enterprise Architecture Governance strategy
- A comprehensive set of Architecture Principles (see Chapter 20) should be established, to guide, inform, and support the way in which an organization sets about fulfilling its mission through the use of IT
- An Architecture Compliance (see Chapter 42) strategy should be adopted — specific measures (more than just a statement of policy) to ensure compliance with the architecture, including Project Impact Assessments, a formal Architecture Compliance review process, and possibly including the involvement of the architecture team in product procurement

Chapter 45

Architecture Maturity Models

This chapter provides techniques for evaluating and quantifying an organization's maturity in Enterprise Architecture.

45.1 Overview

Organizations that can manage change effectively are generally more successful than those that cannot. Many organizations know that they need to improve their processes in order to successfully manage change, but don't know how. Such organizations typically either spend very little on process improvement, because they are unsure how best to proceed; or spend a lot, on a number of parallel and unfocused efforts, to little or no avail.

Capability Maturity Models (CMMs) address this problem by providing an effective and proven method for an organization to gradually gain control over and improve its change processes. Such models provide the following benefits:

- They describe the practices that any organization must perform in order to improve its processes
- They provide a yardstick against which to periodically measure improvement
- They constitute a proven framework within which to manage the improvement efforts
- They organize the various practices into levels, each level representing an increased ability to control and manage the development environment

An evaluation of the organization's practices against the model — called an "assessment" — determines the level at which the organization currently stands. It indicates the organization's ability to execute in the area concerned, and the practices on which the organization needs to focus in order to see the greatest improvement and the highest return on investment. The benefits of CMMs to effectively direct effort are well documented.

45.2 Background

The Software Engineering Institute (SEI) — www.sei.cmu.edu operated by Carnegie Mellon University — developed the original CMM (Capability Maturity Model) for Software (SWCMM) in the early 1990s, which is still widely used today. This CMM provided a framework to develop maturity models in a wide range of disciplines.

The increasing interest in applying these techniques to other fields has resulted in a series of template tools which assess:

- The state of the architecture processes

- The architecture
- The organization's buy-in to both

The main issues addressed by these models include:

- Process implementation and audit
- Quality measurements
- People competencies
- Investment management

They involve use of a multiplicity of models, and focus in particular on measuring business benefits and return on investment.

A closely related topic is the Architecture Skills Framework (see Chapter 46), which can be used to plan the target skills and capabilities required by an organization to successfully develop and utilize Enterprise Architecture, and to determine the training and development needs of individuals.

45.3 US DoC ACMM Framework

45.3.1 Overview

As an example of the trend towards increased interest in applying CMM techniques to Enterprise Architecture, all US Federal agencies are now expected to provide maturity models and ratings as part of their IT investment management and audit requirements.

In particular, the US Department of Commerce (DoC) has developed an Architecture Capability Maturity Model (ACMM)[9] to aid in conducting internal assessments. The ACMM provides a framework that represents the key components of a productive Enterprise Architecture process. The goal is to enhance the overall odds for success of Enterprise Architecture by identifying weak areas and providing a defined evolutionary path to improving the overall architecture process.

The ACMM comprises three sections:

1. The Enterprise Architecture maturity model
2. Enterprise Architecture characteristics of operating units' processes at different maturity levels
3. The Enterprise Architecture CMM scorecard

The first two sections explain the Architecture Capability maturity levels and the corresponding Enterprise Architecture element and characteristics for each maturity level to be used as measures in the assessment process. The third section is used to derive the Architecture Capability maturity level that is to be reported to the DoC Chief Information Officer (CIO).

9. Refer to ocio.os.doc.gov/ITPolicyandPrograms/Enterprise_Architecture/PROD01_004935.

45.3.2 Elements of the ACMM

The DoC ACMM consists of six maturity levels and nine architecture elements. The six levels are:

0 None

1 Initial

2 Under development

3 Defined

4 Managed

5 Measured

The nine Enterprise Architecture elements are:

1 Architecture process

2 Architecture development

3 Business linkage

4 Senior management involvement

5 Operating unit participation

6 Architecture communication

7 IT security

8 Architecture Governance

9 IT investment and acquisition strategy

Two complementary methods are used in the ACMM to calculate a maturity rating. The first method obtains a weighted mean Enterprise Architecture maturity level. The second method shows the percentage achieved at each maturity level for the nine architecture elements.

45.3.3 Example: Enterprise Architecture Process Maturity Levels

The following example shows the detailed characteristics of the Enterprise Architecture maturity levels as applied to each of the nine elements. For example, Level 3: Defined, point number 8 (Explicit documented governance of majority of IT investments) shows Maturity Level 3's state for Element 8 (Architecture Governance).

Level 0: None

No Enterprise Architecture program. No Enterprise Architecture to speak of.

Level 1: Initial

Informal Enterprise Architecture process underway.

1. Processes are *ad hoc* and localized. Some Enterprise Architecture processes are defined. There is no unified architecture process across technologies or business processes. Success depends on individual efforts.
2. Enterprise Architecture processes, documentation, and standards are established by a variety of *ad hoc* means and are localized or informal.

3. Minimal, or implicit linkage to business strategies or business drivers.
4. Limited management team awareness or involvement in the architecture process.
5. Limited operating unit acceptance of the Enterprise Architecture process.
6. The latest version of the operating unit's Enterprise Architecture documentation is on the web. Little communication exists about the Enterprise Architecture process and possible process improvements.
7. IT security considerations are *ad hoc* and localized.
8. No explicit governance of architectural standards.
9. Little or no involvement of strategic planning and acquisition personnel in the Enterprise Architecture process. Little or no adherence to existing standards.

Level 2: Under Development

Enterprise Architecture process is under development.

1. Basic Enterprise Architecture process is documented based on OMB Circular A-130 and Department of Commerce Enterprise Architecture Guidance. The architecture process has developed clear roles and responsibilities.
2. IT vision, principles, business linkages, Baseline, and Target Architecture are identified. Architecture standards exist, but not necessarily linked to Target Architecture. Technical Reference Model (TRM) and Standards Profile framework established.
3. Explicit linkage to business strategies.
4. Management awareness of architecture effort.
5. Responsibilities are assigned and work is underway.
6. The DoC and operating unit Enterprise Architecture web pages are updated periodically and are used to document architecture deliverables.
7. IT security architecture has defined clear roles and responsibilities.
8. Governance of a few architectural standards and some adherence to existing Standards Profile.
9. Little or no formal governance of IT investment and acquisition strategy. Operating unit demonstrates some adherence to existing Standards Profile.

Level 3: Defined

Defined Enterprise Architecture including detailed written procedures and TRM.

1. The architecture is well defined and communicated to IT staff and business management with operating unit IT responsibilities. The process is largely followed.
2. Gap analysis and Migration Plan are completed. Fully developed TRM and Standards Profile. IT goals and methods are identified.
3. Enterprise Architecture is integrated with capital planning and investment control.
4. Senior management team aware of and supportive of the enterprise-wide architecture process. Management actively supports architectural standards.
5. Most elements of operating unit show acceptance of or are actively participating in the Enterprise Architecture process.

6. Architecture documents updated regularly on DoC Enterprise Architecture web page.
7. IT security architecture Standards Profile is fully developed and is integrated with Enterprise Architecture.
8. Explicit documented governance of majority of IT investments.
9. IT acquisition strategy exists and includes compliance measures to IT Enterprise Architecture. Cost benefits are considered in identifying projects.

Level 4: Managed

Managed and measured Enterprise Architecture process.

1. Enterprise Architecture process is part of the culture. Quality metrics associated with the architecture process are captured.
2. Enterprise Architecture documentation is updated on a regular cycle to reflect the updated Enterprise Architecture. Business, Data, Application, and Technology Architectures defined by appropriate *de jure* and *de facto* standards.
3. Capital planning and investment control are adjusted based on the feedback received and lessons learned from updated Enterprise Architecture. Periodic re-examination of business drivers.
4. Senior management team directly involved in the architecture review process.
5. The entire operating unit accepts and actively participates in the Enterprise Architecture process.
6. Architecture documents are updated regularly, and frequently reviewed for latest architecture developments/standards.
7. Performance metrics associated with IT security architecture are captured.
8. Explicit governance of all IT investments. Formal processes for managing variances feed back into Enterprise Architecture.
9. All planned IT acquisitions and purchases are guided and governed by the Enterprise Architecture.

Level 5: Measured

Continuous improvement of Enterprise Architecture process.

1. Concerted efforts to optimize and continuously improve architecture process.
2. A standards and waivers process is used to improve architecture development process.
3. Architecture process metrics are used to optimize and drive business linkages. Business involved in the continuous process improvements of Enterprise Architecture.
4. Senior management involvement in optimizing process improvements in architecture development and governance.
5. Feedback on architecture process from all operating unit elements is used to drive architecture process improvements.
6. Architecture documents are used by every decision-maker in the organization for every IT-related business decision.

7. Feedback from IT security architecture metrics are used to drive architecture process improvements.
8. Explicit governance of all IT investments. A standards and waivers process is used to make governance-process improvements.
9. No unplanned IT investment or acquisition activity.

45.4 Capability Maturity Models Integration (CMMI)

45.4.1 Introduction

The capability models that the SEI is currently involved in developing, expanding, or maintaining include the following:

- CMMI (Capability Maturity Model Integration)
- IPD-CMM® (Integrated Product Development Capability Maturity Model)
- P-CMM® (People Capability Maturity Model)
- SA-CMM® (Software Acquisition Capability Maturity Model)
- SE-CMM® (Systems Engineering Capability Maturity Model)
- SW-CMM® (Capability Maturity Model for Software)

As explained in this chapter, in recent years the industry has witnessed significant growth in the area of maturity models. The multiplicity of models available has led to problems of its own, in terms of how to integrate all the different models to produce a meaningful metric for overall process maturity.

In response to this need, the SEI has developed a Framework called Capability Maturity Model Integration (CMMI), to provide a means of managing the complexity.

According to the SEI, the use of the CMMI models improves on the best practices of previous models in many important ways, in particular enabling organizations to:

- More explicitly link management and engineering activities to business objectives
- Expand the scope of and visibility into the product lifecycle and engineering activities to ensure that the product or service meets customer expectations
- Incorporate lessons learned from additional areas of best practice (e.g., measurement, risk management, and supplier management)
- Implement more robust high-maturity practices
- Address additional organizational functions critical to its products and services
- More fully comply with relevant ISO standards

CMMI is being adopted worldwide.

45.4.2 SCAMPI Method

The Standard CMMI Appraisal Method for Process Improvement (SCAMPI®) is the appraisal method associated with CMMI. The SCAMPI appraisal method is used to identify strengths, weaknesses, and ratings relative to CMMI reference models. It incorporates best practices found successful in the appraisal community, and is based on the features of several legacy appraisal methods. It is applicable to a wide range of appraisal usage modes, including both internal process improvement and external capability determinations.

The SCAMPI method definition document[10] describes the requirements, activities, and practices associated with each of the processes that compose the SCAMPI method.

45.5 Conclusions

This section has sought to introduce into the TOGAF standard the topic of CMM-based methods and techniques for use in relation to Enterprise Architecture.

The benefits of using CMMs are well documented. Future versions of the TOGAF standard may include a maturity model to measure adoption of the TOGAF standard itself.

10. Available at www.sei.cmu.edu/publications/documents/01.reports/01hb001.html.

Chapter 46

Architecture Skills Framework

This chapter provides a set of role, skill, and experience norms for staff undertaking Enterprise Architecture work.

46.1 Introduction

Skills frameworks provide a view of the competency levels required for specific roles. They define:

- The roles within a work area
- The skills required by each role
- The depth of knowledge required to fulfil the role successfully

They are relatively common for defining the skills required for a consultancy and/or project management assignment, to deliver a specific project or work package. They are also widely used by recruitment and search agencies to match candidates and roles.

Their value derives from their ability to provide a means of rapidly identifying skill matches and gaps. Successfully applied, they can ensure that candidates are fit for the jobs assigned to them.

Their value in the context of Enterprise Architecture arises from the immaturity of the Enterprise Architecture discipline, and the problems that arise from this.

46.2 Need for an Enterprise Architecture Skills Framework

46.2.1 Definitional Rigor

"Enterprise Architecture" and "Enterprise Architect" are widely used but poorly defined terms in industry today. They are used to denote a variety of practices and skills applied in a wide variety of architecture domains. There is a need for better classification to enable more implicit understanding of what type of architecture/architect is being described.

This lack of uniformity leads to difficulties for organizations seeking to recruit or assign/promote staff to fill positions in the architecture field. Because of the different usage of terms, there is often misunderstanding and miscommunication between those seeking to recruit for, and those seeking to fill, the various roles of the architect.

46.2.2 Basis of an Internal Architecture Practice

Despite the lack of uniform terminology, architecture skills are in increasing demand, as the discipline of architecture gains increasing attention within industry.

Many enterprises have set up, or are considering setting up, an Enterprise Architecture practice, as a means of fostering development of the necessary skills and experience among in-house staff to undertake the various architecting tasks required by the enterprise.

An Enterprise Architecture practice is a formal program of development and certification, by which an enterprise formally recognizes the skills of its practicing architects, as demonstrated by their work. Such a program is essential in order to ensure the alignment of staff skills and experience with the architecture tasks that the enterprise wishes to be performed.

The role and skill definitions on which such a program needs to be based are also required, by both recruiting and supplying organizations, in cases where external personnel are to be engaged to perform architecture work (for example, as part of a consultancy engagement).

An Enterprise Architecture practice is both difficult and costly to set up. It is normally built around a process of peer review, and involves the time and talent of the strategic technical leadership of an enterprise. Typically it involves establishment of a peer review board, and documentation of the process, and of the requirements for internal certification. Time is also required of candidates to prepare for peer review, by creating a portfolio of their work to demonstrate their skills, experiences, and contributions to the profession.

The TOGAF Architecture Skills Framework attempts to address this need by providing definitions of the architecting skills and proficiency levels required of personnel, internal or external, who are to perform the various architecting roles defined within the TOGAF framework.

Because of the complexity, time, and cost involved, many enterprises do not have an internal Enterprise Architect certification program, preferring instead to simply interview and recruit architecture staff on an *ad hoc* basis. There are serious risks associated with this approach:

- Communication between recruiting organizations, consultancies, and employment agencies is very difficult
- Time is wasted interviewing staff who may have applied in all good faith, but still lack the skills and/or experience required by the employer
- Staff that are capable of filling architecture roles may be overlooked, or may not identify themselves with advertised positions and hence not even apply
- There is increased risk of unsuitable personnel being employed or engaged, through no-one's fault, and despite everyone involved acting in good faith

  This in turn can:

  — Increase personnel costs, through the need to rehire or reassign staff

  — Adversely impact the time, cost, and quality of operational IT systems, and the projects that deliver them

46.3 Goals/Rationale

46.3.1 Certification of Enterprise Architects

The main purpose behind an enterprise setting up an internal Enterprise Architect certification program is two-fold:

1. To formally recognize the skill of its practicing architects, as part of the task of establishing and maintaining a professional architecting organization
2. To ensure the alignment of necessary staff skills and experience with the architecture tasks that the enterprise wishes to be performed, whether these are to be performed internally to the enterprise or externally; for example, as part of a consultancy engagement

46.3.2 Specific Benefits

Specific benefits anticipated from use of the TOGAF Architecture Skills Framework include:

- Reduced time, cost, and risk in training, hiring, and managing architecture professionals, both internal and external:
  - Simplifies communication between recruiting organizations, consultancies, and employment agencies
  - Avoids wasting time interviewing staff who may have applied in all good faith, but still lack the skills and/or experience required by the employer
  - Avoids staff who are capable of filling architecture roles being overlooked, or not identifying themselves with advertised positions and hence not even applying
- Reduced time and cost to set up an internal architecture practice:
  - Many enterprises do not have an internal architecture practice due to the complexity involved in setting one up, preferring instead to simply interview and recruit architecture staff on an *ad hoc* basis
  - By providing definitions of the architecting skills and proficiency levels required of personnel who are to perform the various architecting roles defined within the TOGAF standard, the Architecture Skills Framework greatly reduces the time, cost, and risk of setting up a practice for the first time, and avoids "re-inventing wheels"
  - Enterprises that already have an internal architecture practice are able to set enterprise-wide norms, but still experience difficulties as outlined above in recruiting staff, or engaging consultants, from external sources, due to the lack of uniformity between different enterprises

    By aligning its existing skills framework with the industry-accepted definitions provided by The Open Group, an enterprise can greatly simplify these problems.
- Reduced time and cost to implement an architecture practice helps reduce the time, cost, and risk of overall solution development:
  - Enterprises that do not have an internal architecture practice run the risk of unsuitable personnel being employed or engaged, through no-one's fault, and despite everyone involved acting in good faith

    The resultant time and cost penalties far outweigh the time and cost of having an internal architecture practice:

- Personnel costs are increased, through the occasional need to rehire or reassign staff
- Even more important is the adverse impact on the time, cost, and quality of operational IT systems, and the projects to deliver them, resulting from poor staff assignments

46.4 Enterprise Architecture Role and Skill Categories

46.4.1 Overview

This section describes the role of an Enterprise Architect, the fundamental skills required, and some possible disciplines in which an Enterprise Architect might specialize.

The TOGAF standard delivers an Enterprise Architecture, and therefore requires both business and IT-trained professionals to develop the Enterprise Architecture.

The TOGAF Architecture Skills Framework provides a view of the competency levels for specific roles within the Enterprise Architecture team. The Framework defines:

- The roles within an Enterprise Architecture work area
- The skills required by those roles
- The depth of knowledge required to fulfil each role successfully

The value is in providing a rapid means of identifying skills and gaps. Successfully applied, the Framework can be used as a measure for:

- Staff development
- Ensuring that the right person does the right job

46.4.2 TOGAF Roles

A typical architecture team undertaking the development of an Enterprise Architecture as described in the TOGAF standard would comprise the following roles:

- Architecture Board Members
- Architecture Sponsor
- Architecture Manager
- Architects for:
  - Enterprise Architecture (which for the purpose of the tables shown below can be considered as a superset of Business, Data, Application, and Technology Architecture)
  - Business Architecture
  - Data Architecture
  - Application Architecture
  - Technology Architecture
- Program and/or Project Managers

- IT Designer
- And many others . . .

The tables that follow show, for each of these roles, the skills required and the desirable level of proficiency in each skill.

Of all the roles listed above, the one that needs particularly detailed analysis and definition is of course the central role of Enterprise Architect. As explained above, "Enterprise Architecture" and "Enterprise Architect" are terms that are very widely used but very poorly defined in industry today, denoting a wide variety of practices and skills applied in a wide variety of architecture domains. There is often confusion between the role of an architect and that of a designer or builder. Many of the skills required by an Enterprise Architect are also required by the designer, who delivers the solutions. While their skills are complementary, those of the designer are primarily technology-focused and translate the architecture into deliverable components.

The final subsection below therefore explores in some detail the generic characteristics of the role of Enterprise Architect, and the key skill requirements, whatever the particular architecture domain (Enterprise Architecture, Business Architecture, Data Architecture, Application Architecture, Technology Architecture, etc.).

46.4.3 Categories of Skills

The TOGAF team skill set will need to include the following main categories of skills:

- **Generic Skills**: — typically comprising leadership, teamworking, inter-personal skills, etc.
- **Business Skills & Methods**: — typically comprising business cases, business process, strategic planning, etc.
- **Enterprise Architecture Skills**: — typically comprising modeling, building block design, applications and role design, systems integration, etc.
- **Program or Project Management Skills**: — typically comprising managing business change, project management methods and tools, etc.
- **IT General Knowledge Skills**: — typically comprising brokering applications, asset management, migration planning, SLAs, etc.
- **Technical IT Skills**: — typically comprising software engineering, security, data interchange, data management, etc.
- **Legal Environment**: — typically comprising data protection laws, contract law, procurement law, fraud, etc.

The tables that follow illustrate each of these categories of skills.

The tables that follow show, for each of these skills, the roles to which they are relevant and the desirable level of proficiency in each skill.

46.4.4 Proficiency Levels

The TOGAF Architecture Skills Framework identifies four levels of knowledge or proficiency in any area:

Level	Achievement	Description
1	Background	Not a required skill, though should be able to define and manage skill if required.
2	Awareness	Understands the background, issues, and implications sufficiently to be able to understand how to proceed further and advise client accordingly.
3	Knowledge	Detailed knowledge of subject area and capable of providing professional advice and guidance. Ability to integrate capability into architecture design.
4	Expert	Extensive and substantial practical experience and applied knowledge on the subject.

46.5 Enterprise Architecture Role and Skill Definitions

46.5.1 Generic Skills

Roles	Architecture Board Member	Architecture Sponsor	Enterprise Architecture Manager	Enterprise Architecture Technology	Enterprise Architecture Data	Enterprise Architecture Applications	Enterprise Architecture Business	Program/ Project Manager	IT Designer
Generic Skills									
Leadership	4	4	4	3	3	3	3	4	1
Teamwork	3	3	4	4	4	4	4	4	2
Inter-personal	4	4	4	4	4	4	4	4	2
Oral Communications	3	3	4	4	4	4	4	4	2
Written Communications	3	3	4	4	4	4	4	3	3
Logical Analysis	2	2	4	4	4	4	4	3	3
Stakeholder Management	4	3	4	3	3	3	3	4	2
Risk Management	3	3	4	3	3	3	3	4	1

46.5.2 Business Skills & Methods

Roles	Architecture Board Member	Architecture Sponsor	Enterprise Architecture Manager	Enterprise Architecture Technology	Enterprise Architecture Data	Enterprise Architecture Applications	Enterprise Architecture Business	Program/ Project Manager	IT Designer
Business Skills & Methods									
Business Case	3	4	4	4	4	4	4	4	2
Business Scenario	2	3	4	4	4	4	4		2
Organization	3	3	4	3	3	3	4	3	2
Business Process	3	3	4	4	4	4	4	3	2
Strategic Planning	2	3	3	3	3	3	4	3	1
Budget Management	3	3	3	3	3	3	3	4	3
Visioning	3	3	4	3	3	3	4	3	2
Business Metrics	3	4	4	4	4	4	4	4	3
Business Culture	4	4	4	3	3	3	3	3	1
Legacy Investments	4	4	3	2	2	2	2	3	2
Business Functions	3	3	3	3	4	4	4	3	2

46.5.3 Enterprise Architecture Skills

Roles	Architecture Board Member	Architecture Sponsor	Enterprise Architecture Manager	Enterprise Architecture Technology	Enterprise Architecture Data	Enterprise Architecture Applications	Enterprise Architecture Business	Program/ Project Manager	IT Designer
Enterprise Architecture Skills									
Business Modeling	2	2	4	3	3	4	4	2	2
Business Process Design	1	1	4	3	3	4	4	2	2
Role Design	2	2	4	3	3	4	4	2	2
Organization Design	2	2	4	3	3	4	4	2	2
Data Design	1	1	3	3	4	3	3	2	3
Application Design	1	1	3	3	3	4	3	2	3
Systems Integration	1	1	4	4	3	3	3	2	2
IT Industry Standards	1	1	4	4	4	4	3	2	3
Services Design	2	2	4	4	3	4	3	2	2
Architecture Principles Design	2	2	4	4	4	4	4	2	2
Architecture Views & Viewpoints Design	2	2	4	4	4	4	4	2	2
Building Block Design	1	1	4	4	4	4	4	2	3
Solutions Modeling	1	1	4	4	4	4	4	2	3
Benefits Analysis	2	2	4	4	4	4	4	4	2
Business Interworking	3	3	4	3	3	4	4	3	1
Systems Behavior	1	1	4	4	4	4	3	3	2
Project Management	1	1	3	3	3	3	3	4	2

46.5.4 Program or Project Management Skills

Roles	Architecture Board Member	Architecture Sponsor	Enterprise Architecture Manager	Enterprise Architecture Technology	Enterprise Architecture Data	Enterprise Architecture Applications	Enterprise Architecture Business	Program/ Project Manager	IT Designer
Program or Project Management Skills									
Program Management	1	2	3	3	3	3	3	4	2
Project Management	1	2	3	3	3	3	3	4	2
Managing Business Change	3	3	4	3	3	3	4	4	2
Change Management	3	3	4	3	3	3	4	3	2
Value Management	4	4	4	3	3	3	4	3	2

46.5.5 IT General Knowledge Skills

Roles	Architecture Board Member	Architecture Sponsor	Enterprise Architecture Manager	Enterprise Architecture Technology	Enterprise Architecture Data	Enterprise Architecture Applications	Enterprise Architecture Business	Program/ Project Manager	IT Designer
IT General Knowledge Skills									
IT Application Development Methodologies & Tools	2	2	3	4	4	4	2	3	3
Programming Languages	1	1	3	4	4	4	3	2	3
Brokering Applications	1	1	3	3	4	4	3	2	3
Information Consumer Applications	1	1	3	3	4	4	3	2	3
Information Provider Applications	1	1	3	3	4	4	3	2	3
Storage Management	1	1	3	4	4	2	2	2	3
Networks	1	1	3	4	3	2	2	2	3
Web-based Services	1	1	3	3	4	4	2	2	3
IT Infrastructure	1	1	3	4	3	2	2	2	3
Asset Management	1	1	4	4	3	3	3	2	3
Service Level Agreements	1	1	4	4	3	4	3	2	3
Systems	1	1	3	4	3	3	2	2	3
COTS	1	1	3	4	3	4	2	2	3
Enterprise Continuums	1	1	4	4	4	4	4	2	3
Migration Planning	1	1	4	3	4	3	3	2	3
Management Utilities	1	1	3	2	4	4	2	2	3
Infrastructure	1	1	3	4	3	4	2	2	3

46.5.6 Technical IT Skills

Roles	Architecture Board Member	Architecture Sponsor	Enterprise Architecture Manager	Enterprise Architecture Technology	Enterprise Architecture Data	Enterprise Architecture Applications	Enterprise Architecture Business	Program/ Project Manager	IT Designer
Technical IT Skills									
Software Engineering	1	1	3	3	4	4	3	2	3
Security	1	1	3	4	3	4	3	2	3
Systems & Network Management	1	1	3	4	3	3	3	2	3
Transaction Processing	1	1	3	4	3	4	3	2	3
Location & Directory	1	1	3	4	4	3	3	2	3
User Interface	1	1	3	4	4	4	3	2	3
International Operations	1	1	3	4	3	3	2	2	2
Data Interchange	1	1	3	4	4	3	2	2	3
Data Management	1	1	3	4	4	3	2	2	3
Graphics & Image	1	1	3	4	3	3	2	2	3
Operating System Services	1	1	3	4	3	3	2	2	3
Network Services	1	1	3	4	3	3	2	2	3
Communications Infrastructure	1	1	3	4	3	3	2	2	3

46.5.7 Legal Environment

Roles	Architecture Board Member	Architecture Sponsor	Enterprise Architecture Manager	Enterprise Architecture Technology	Enterprise Architecture Data	Enterprise Architecture Applications	Enterprise Architecture Business	Program/ Project Manager	IT Designer
Legal Environment									
Contract Law	2	2	2	2	2	2	2	3	1
Data Protection Law	3	3	4	3	3	3	3	2	2
Procurement Law	3	2	2	2	2	2	2	4	1
Fraud	3	3	3	3	3	3	3	3	1
Commercial Law	3	3	2	2	2	2	3	3	1

46.6 Generic Role and Skills of the Enterprise Architect

Of all the roles listed above, the one that needs particularly detailed analysis and definition is, of course, the central role of Enterprise Architect. As explained above, "Enterprise Architecture" and "Enterprise Architect" are terms that are very widely used but very poorly defined in industry today, denoting a wide variety of practices and skills applied in a wide variety of architecture domains.

This section therefore explores in some detail the generic characteristics of the role of Enterprise Architect, and some key skill requirements, whatever the particular architecture domain (Enterprise Architecture, Business Architecture, Data Architecture, Application Architecture, Technology Architecture, etc.).

46.6.1 Generic Role

Enterprise Architects are visionaries, coaches, team leaders, business-to-technical liaisons, computer scientists, and industry experts.

The following is effectively a job description for an Enterprise Architect:

> "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 of which particular architecture views to develop is one of the key decisions that the Enterprise Architect has to make. 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)."

The role of the Enterprise Architect is more like that of a city planner than that of a building architect, and the product of the Enterprise Architect is more aptly characterized as a planned community (as opposed to an unconstrained urban sprawl), rather than as a well-designed building or set of buildings.

An Enterprise Architect does not create the technical vision of the enterprise, but has professional relationships with executives of the enterprise to gather and articulate the technical vision, and to produce the strategic plan for realizing it. This plan is always tied to the business plans of the enterprise, and design decisions are traceable to the business plan.

The strategic plan of the Enterprise Architect is tied to the Architecture Governance process (see Chapter 44) for the enterprise, so design decisions are not circumvented for tactical convenience.

The Enterprise Architect produces documentation of design decisions for application development teams or product implementation teams to execute.

An architect is involved in the entire process; beginning with working with the customer to understand real needs, as opposed to wants, and then throughout the process to translate those needs into capabilities verified to meet the needs. Additionally, the architect may present different models to the customer that communicate how those needs may be met, and is therefore an essential participant in the consultative selling process.

However, the architect is not the builder, and must remain at a level of abstraction necessary to ensure that they do not get in the way of practical implementation.

The following excerpt from *The Art of Systems Architecting* (Rechtin and Maier, 2000) depicts this notion:

> "It is the responsibility of the architect to know and concentrate on the critical few details and interfaces that really matter, and not to become overloaded with the rest."

The architect's focus is on understanding what it takes to satisfy the client, where qualitative worth is used more than quantitative measures. The architect uses more inductive skills than the deductive skills of the builder. The architect deals more with guidelines, rather than rules that builders use as a necessity.

It also must be clear that the role of an architect may be performed by an engineer. A goal of this document is to describe the role — what should be done, regardless of who is performing it.

Thus, the role of the architect can be summarized as to:

- **Understand and interpret requirements**: probe for information, listen to information, influence people, facilitate consensus building, synthesize and translate ideas into actionable requirements, articulate those ideas to others, and identify use or purpose, constraints, risks, etc.

  The architect participates in the discovery and documentation of the customer's business scenarios that are driving the solution. The architect is responsible for requirements understanding and embodies that requirements understanding in the architecture specification.

- **Create a useful model**: take the requirements and develop well-formulated models of the components of the solution, augmenting the models as necessary to fit all of the circumstances, and show multiple views through models to communicate the ideas effectively

  The architect is responsible for the overall architecture integrity and maintaining the vision of the offering from an architectural perspective. The architect also ensures leverage opportunities are identified, using building blocks, and is a liaison between the functional groups (especially development and marketing) to ensure that the leverage opportunities are realized. The architect provides and maintains these models as a framework for understanding the domain(s) of development work, guiding what should be done within the organization, or outside the organization. The architect must represent the organization view of the architecture by understanding all the necessary business components.

- **Validate, refine, and expand the model**: verify assumptions, bring in subject matter experts, etc. in order to improve the model and to further define it, adding as necessary new ideas to make the result more flexible and more tightly linked to current and expected requirements

  The architect additionally should assess the value of solution-enhancing developments emanating from field work and incorporate these into the architecture models as appropriate.

- **Manage the architecture**: continuously monitor the models and update them as necessary to show changes, additions, and alterations

  Represent architecture and issues during development and decision points of the program. The architect is an "agent of change", representing that need for the implementation of the architecture. Through this development cycle, the architect continuously fosters the sharing of customer, architecture, and technical information between organizations.

46.6.2 Characterization in Terms of the Enterprise Continuum

Under certain circumstances, the complexity of a solution may require additional architects to support the architecture effort. The different categories of architects are described below, but as they are architects, they all perform the tasks described above. Any combination of Enterprise, Enterprise Solution, and Solution Architects may be utilized, as a team. In such cases each member may have a specific focus, if not specific roles and responsibilities, within the phases of the development process. In cases where a team of architects is deemed necessary, a lead Enterprise Architect should be assigned to manage and lead the team members.

- The **Enterprise Architect** has the responsibility for architectural design and documentation at a landscape and technical reference model level

  The Enterprise Architect often leads a group of the Segment Architects and/or Solution Architects related to a given program. The focus of the Enterprise Architect is on enterprise-level business functions required.

- The **Segment Architect** has the responsibility for architectural design and documentation of specific business problems or organizations

  A Segment Architect re-uses the output from all other architects, joining detailed technical solutions to the overall architectural landscape. The focus of the Segment Architect is on enterprise-level business solutions in a given domain, such as finance, human resources, sales, etc.

- The **Solution Architect** has the responsibility for architectural design and documentation at a system or subsystem level, such as management or security

  A Solution Architect may shield the Enterprise/Segment Architect from the unnecessary details of the systems, products, and/or technologies. The focus of the Solution Architect is on system technology solutions; for example, a component of a solution such as enterprise data warehousing.

46.6.3 Key Characteristics of an Enterprise Architect

46.6.3.1 Skills and Experience in Producing Designs

An Enterprise Architect must be proficient in the techniques that go into producing designs of complex systems, including requirements discovery and analysis, formulation of solution context, identification of solution alternatives and their assessment, technology selection, and design configuration.

46.6.3.2 Extensive Technical Breadth, with Technical Depth in One or a Few Disciplines

An Enterprise Architect should possess an extensive technical breadth through experience in the IT industry. This breadth should be in areas of application development and deployment, and in the areas of creation and maintenance of the infrastructure to support the complex application environment. Current IT environments are heterogeneous by nature, and the experienced Enterprise Architect will have skills across multiple platforms, including distributed systems and traditional mainframe environments. Enterprise Architects will have, as a result of their careers, skills in at least one discipline that is considered to be at the level of a subject matter expert.

46.6.3.3 Method-Driven Approach to Execution

Enterprise Architects approach their job through the consistent use of recognized design methods such as the TOGAF Architecture Development Method (ADM). Enterprise Architects should have working knowledge of more than one design method and be comfortable deploying parts of methods appropriate to the situation in which they are working. This should be seen in the body of design work the Enterprise Architect has produced through repeated successful use of more than one design method. Proficiency in methodology use is in knowing what parts of methods to use in a given situation, and what methods not to use.

46.6.3.4 Full Project Scope Experience

While Enterprise Architects are responsible for design and hand-off of the project to implementors, it is vital that they have experience with all aspects of a project from design through development, testing, implementation, and production. This scope of experience will serve to keep Enterprise Architects grounded in the notion of fitness-for-purpose and the practical nature of system implementation. The impact of full project scope experience should lead the Enterprise Architect to make better design decisions, and better inform the trade-offs made in those decisions.

46.6.3.5 Leadership

Communication and team building are key to the successful role of the Enterprise Architect. The mix of good technical skill and the ability to lead are crucial to the job. The Enterprise Architect should be viewed as a leader in the enterprise by the IT organization, the clients they serve, and management.

46.6.3.6 Personal and Professional Skills

The Enterprise Architect must have strong communications and relationship skills. A major task of the Enterprise Architect is to communicate complex technical information to all stakeholders of the project, including those who do not have a technical background. Strong negotiation and problem-solving skills are also required. The Enterprise Architect must work with the project management team to make decisions in a timely manner to keep projects on track.

46.6.3.7 Skills and Experience in One or More Industries

Industry skill and experience will make the task of gathering requirements and deciding priorities easier and more effective for the Enterprise Architect. Enterprise Architects must understand the business processes of the enterprise in which they work, and how those processes work with other peer enterprises in the industry. They should also be able to spot key trends and correct flawed processes, giving the IT organization the capability to lead the enterprise, not just respond to requests. The mission of the Enterprise Architect is strategic technical leadership.

46.7 Conclusions

The TOGAF Architecture Skills Framework provides an assessment of the skills required to deliver a successful Enterprise Architecture.

It is hoped that the provision of this Architecture Skills Framework will help reduce the time, cost, and risk involved in training, recruiting, and managing IT architecture professionals, and at the same time enable and encourage more organizations to institute an internal IT architecture practice, hopefully based on (or at least leveraging) the role and skill definitions provided.

The TOGAF Standard, Version 9.2

Part VII:

Appendices

The Open Group

Appendix A

Glossary of Supplementary Definitions

This appendix contains additional definitions to supplement the definitions contained in Chapter 3.

A.1 Application Software

Software entities which have a specific business purpose.

A.2 Availability

In the context of IT systems, the probability that system functional capabilities are ready for use by a user at any time, where all time is considered, including operations, repair, administration, and logistic time. Availability is further defined by system category for both routine and priority operations.

A.3 Business System

Hardware, software, policy statements, processes, activities, standards, and people which together implement a business function.

A.4 Catalog

A structured list of architectural outputs of a similar kind, used for reference. For example, a technology standards catalog or an application portfolio.

A.5 Client

An application component which requests services from a server.

A.6 COBIT

An acronym for Control OBjectives for Information and related Technology, created by the Information Systems Audit and Control Association (ISACA) and the IT Governance Institute (ITGI), which provides a set of recommended best practices for the governance/management of information systems and technology.

A.7 Configuration Management

A discipline applying technical and administrative direction and surveillance to:

- Identify and document the functional and physical characteristics of a configuration item
- Control changes to those characteristics
- Record and report changes to processing and implementation status

Also, the management of the configuration of Enterprise Architecture practice (intellectual property) assets and baselines and the control of change over of those assets.

A.8 Contract

An agreement between a service consumer and a service provider that establishes functional and non-functional parameters for interaction.

A.9 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.

A.10 CxO

The chief officer within a particular function of the business; e.g., Chief Executive Officer, Chief Financial Officer, Chief Information Officer, Chief Technology Officer.

A.11 Data Dictionary

A specialized type of database containing metadata; a repository of information describing the characteristics of data used to design, monitor, document, protect, and control data in information systems and databases; an application system supporting the definition and management of database metadata.

A.12 Data Element

A basic unit of information having a meaning and that may have subcategories (data items) of distinct units and values.

A.13 Data Entity

An encapsulation of data that is recognized by a business domain expert as a thing. Logical data entities can be tied to applications, repositories, and services and may be structured according to implementation considerations.

A.14 Database

A structured or organized collection of data entities, which is to be accessed by a computer.

A.15 Database Management System

A computer application program that accesses or manipulates the database.

A.16 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.

A.17 End User

Person who ultimately uses the computer application or output.

A.18 Enterprise Resource Planning (ERP) System

A complete suite of integrated applications that support the major business support functions of an organization; e.g., Financial (AP/AR/GL), HR, Payroll, Stock, Order Processing and Invoicing, Purchasing, Logistics, Manufacturing, etc.

A.19 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.

A.20 Functional Decomposition

A hierarchy of the functions of an enterprise or organization.

A.21 Goal

A high-level statement of intent or direction for an organization. Typically used to measure success of an organization.

A.22 Guideline

An architectural document that provides guidance on the optimal ways to carry out design or implementation activities.

A.23 Hardware

The physical infrastructure needed to run software; e.g., servers, workstations, network equipment, etc.

A.24 Information Domain

Grouping of information (or data entities) by a set of criteria such as security classification, ownership, location, etc. In the context of security, information domains are defined as a set of users, their information objects, and a security policy.

A.25 Information System (IS)

The computer (or IT)-based portion of a business system.

A.26 Interaction

A relationship between architectural building blocks (i.e., services or components) that embodies communication or usage.

A.27 Interaction Model

An architectural view, catalog, or matrix that shows a particular type of interaction. For example, a diagram showing application integration.

A.28 Interface

Interconnection and inter-relationships between, for example, people, systems, devices, applications, or the user and an application or device.

A.29 Key Performance Indicator (KPI)

A way of quantifying the performance of the business or project.

A.30 Lifecycle

The period of time that begins when a system is conceived and ends when the system is no longer available for use.

A.31 Location

A place where business activity takes place and can be hierarchically decomposed.

A.32 Logical Application Component

An encapsulation of application functionality that is independent of a particular implementation. For example, the classification of all purchase request processing applications implemented in an enterprise.

A.33 Logical Data Component

A boundary zone that encapsulates related data entities to form a logical location to be held. For example, external procurement information.

A.34 Logical Technology Component

An encapsulation of technology infrastructure that is independent of a particular product. A class of technology product. For example, supply chain management software as part of an Enterprise Resource Planning (ERP) suite or a Commercial Off-The-Shelf (COTS) purchase request processing enterprise service.

A.35 Managing Successful Programs (MSP)

A best practice methodology for program management, developed by the UK Office of Government Commerce (OGC).

A.36 Matrix

A format for showing the relationship between two (or more) architectural elements in a grid format.

A.37 Measure

An indicator or factor that can be tracked, usually on an ongoing basis, to determine success or alignment with objectives and goals.

A.38 Metaview

A pattern or template of the view, from which to develop individual views. Establishes the purposes and audience for a view, the ways in which the view is documented (e.g., for visual modeling), and the ways in which it is used (e.g., for analysis).

See also Section 3.18 in Chapter 3.

A.39 Open System

A system that implements sufficient open specifications for interfaces, services, and supporting formats to enable properly engineered application software:

- To be ported with minimal changes across a wide range of systems
- To interoperate with other applications on local and remote systems
- To interact with users in a style that facilitates user portability

A.40 Operational Governance

The operational performance of systems against contracted performance levels, the definition of operational performance levels, and the implementation of systems that ensure effective operation of systems.

See also Section 3.43 in Chapter 3.

A.41 Packaged Services

Services that are acquired from the market from a Commercial Off-The-Shelf (COTS) vendor, rather than being constructed via code build.

A.42 Physical Application Component

An application, application module, application service, or other deployable component of functionality. For example, a configured and deployed instance of a Commercial Off-The-Shelf (COTS) Enterprise Resource Planning (ERP) supply chain management application.

A.43 Physical Data Component

A boundary zone that encapsulates related data entities to form a physical location to be held. For example, a purchase order business object, comprising purchase order header and item business object nodes.

A.44 Physical Technology Component

A specific technology infrastructure product or technology infrastructure product instance. For example, a particular product version of a Commercial Off-The-Shelf (COTS) solution, or a specific brand and version of server.

A.45 Portability

1. The ease with which a system, component, data, or user can be transferred from one hardware or software environment to another.
2. A quality metric that can be used to measure the relative effort to transport the software for use in another environment or to convert software for use in another operating environment, hardware configuration, or software system environment.

A.46 Portfolio

The complete set of change activities or systems that exist within the organization or part of the organization. For example, application portfolio and project portfolio.

A.47 PRINCE2

An acronym for PRojects IN Controlled Environments, which is a standard project management method.

A.48 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 function or service (at next level of detail). Processes may also be used to link or compose organizations, functions, services, and processes.

A.49 Product

Output generated by the business. The business product of the execution of a process.

A.50 Profile

A set of one or more base standards and, where applicable, the identification of those classes, subsets, options, and parameters of those base standards, necessary for accomplishing a particular function.

A.51 Profiling

Identifying standards and characteristics of a particular system.

A.52 Program

A co-ordinated set of change projects that deliver business benefit to the organization.

A.53 Project

A single change project which delivers business benefit to the organization.

A.54 Risk Management

The management of risks and issues that may threaten the success of the Enterprise Architecture practice and its ability to meet its vision, goals, and objectives, and, importantly, its service provision.

Note: Risk management is described in Part III, Chapter 27.

A.55 Scalability

The ability to use the same application software on many different classes of hardware/software platforms from PCs to super-computers (extends the portability concept). The capability to grow to accommodate increased work loads.

A.56 Security

Services which protect data, ensuring its confidentiality, availability, and integrity.

A.57 Server

An application component which responds to requests from a client.

A.58 Service Quality

A preset configuration of non-functional attributes that may be assigned to a service or service contract.

A.59 SMART

An acronym for Specific, Measurable, Actionable, Realistic, and Time-bound, which is an approach to ensure that targets and objectives are set in a way that can be achieved and measured.

A.60 Supplier Management

The management of suppliers of products and services to the Enterprise Architecture practice in concert with larger corporate procurement activities.

A.61 System

A combination of interacting elements organized to achieve one or more stated purposes (Source: ISO/IEC/IEEE 15288: 2015).

A.62 Time Period

The timeframe over which the potential impact is to be measured.

A.63 Transaction

Interaction between a user and a computer in which the user inputs a command to receive a specific result from the computer.

A.64 Use-Case

A view of organization, application, or product functionality that illustrates capabilities in context with the user of that capability.

A.65 User

1. Any person, organization, or functional unit that uses the services of an information processing system.
2. In a conceptual schema language, any person or any thing that may issue or receive commands and messages to or from the information system.

Appendix B

Abbreviations

ABB	Architecture Building Block
ACMM	Architecture Capability Maturity Model
ADM	Architecture Development Method
ANSI	American National Standards Institute
API	Application Platform Interface
ARTS	Association for Retail Technology Standards
BMM	Business Motivation Model
BPM	Business Process Management
BPMN	Business Process Modeling Notation
BTEP	The Canadian Government Business Transformation Enablement Program
CMM	Capability Maturity Models
CMMI	Capability Maturity Model Integration
COBIT	Control OBjectives for Information and related Technology
COTS	Commercial Off-The-Shelf applications
CRM	Customer Relationship Management
CRUD	Create/Read/Update/Delete
CSF	Critical Success Factor
DBA	Database Administrator
DBMS	Database Management System
DoC	US Department of Commerce
DoD	US Department of Defense
DoDAF	Department of Defense Architecture Framework
EAI	Enterprise Application Integration
EDIFACT	(United Nations) Electronic Data Interchange For Administration, Commerce, and Transport
ERP	Enterprise Resource Planning
ETL	Extract, Transform, Load
FICO	Fair Isaac Corporation

FTE	Full-Time Equivalent
GOTS	Government Off-The-Shelf applications
HIPAA	Health Insurance Portability and Accountability Act
ICAM	Integrated Computer Aided Manufacturing
ICOM	Inputs, Controls, Outputs, and Mechanisms/Resources
IDEF	Integrated Computer Aided Manufacturing (ICAM) DEFinition
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronic Engineers
III-RM	Integrated Information Infrastructure Reference Model
IPD-CMM	Integrated Product Development Capability Maturity Model
ISACA	Information Systems Audit and Control Association
ISACF	Information Systems Audit and Control Foundation
ISO	International Standards Organization
IT	Information Technology
ITGI	IT Governance Institute
ITIL	Information Technology Infrastructure Library
ITPMF	IT Portfolio Management Facility
J2EE	Java 2 Platform, Enterprise Edition
KPI	Key Performance Indicator
LAN	Local Area Network
MDA	Model-Driven Architecture
MSP	Managing Successful Programs
NASCIO	National Association of State Chief Information Officers
OECD	Organization for Economic Co-operation and Development
OGC	UK Office of Government Commerce
OLA	Operational-Level Agreement
OMB	Office of Management and Budget
OMG	Object Management Group
ORB	Object Request Broker
OSI	Open Systems Interconnection
OSOA	Open Service-Oriented Architecture
P-CMM	People Capability Maturity Model
PDF	Portable Document Format
PMBOK	Project Management Body of Knowledge

PRINCE	PRojects in Controlled Environments
QoS	Quality of Service
RAS	Remote Access Services
RFC	Request For Change
RFI	Request for Information
RFP	Request for Proposal
RM	Reference Model
SA-CMM	Software Acquisition Capability Maturity Model
SBB	Solution Building Block
SCA	Service Component Architecture
SCAMPI	Standard CMMI Appraisal Method for Process Improvement
SDO	Service Data Objects
SE-CMM	Systems Engineering Capability Maturity Model
SEI	Software Engineering Institute
SGML	Standard Generalized Markup Language
SIB	Standards Information Base
SLA	Service-Level Agreement
SMART	Specific, Measurable, Attainable, Realistic, and Time-bound
SOA	Service-Oriented Architecture
SPEM	Software Processing Engineering Metamodel
SW-CMM	Capability Maturity Model for Software
SysML	Systems Modeling Language
TAFIM	Technical Architecture Framework for Information Management
TRM	Technical Reference Model
UML	Unified Modeling Language
WAN	Wide Area Network
XML	Extensible Markup Language

Index