The Open Group Guide

A Practical Approach to Application Portfolio Consolidation using the TOGAF® Standard

The Open Group Guide

A Practical Approach to Application Portfolio Consolidation using the TOGAF® Standard

ISBN: 1-947754-18-8

Document Number: G18B

Published by The Open Group, July 2018.

Comments relating to the material contained in this document may be submitted to:

The Open Group, Apex Plaza, Forbury Road, Reading, Berkshire, RG1 1AX, United Kingdom

or by electronic mail to:

ogspecs@opengroup.org

Contents

Preface

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 600 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.

This Document

This document is The Open Group Guide: A Practical Approach to Application Portfolio Consolidation using the TOGAF® Standard. It has been developed and approved by The Open Group.

This Guide is intended to help enterprises who are attempting to simplify, rationalize, and consolidate their application portfolios in support of delivering business value and enabling business agility. It shows how enterprises can use the TOGAF® standard to identify and deliver opportunities to simplify their application landscape, and to help in the transition from legacy to cloud.

Enterprises can benefit from this Guide as it provides support on how to shape their Application Architecture when consolidating complex application landscapes and introduces approaches and techniques to classify applications based on their value to the organization and their relative health to allow consistent assessments to be made across the application portfolio.

Enterprise and Application Architects can benefit from this Guide as it provides tools to identify and reduce duplication and overlaps in application functionality, manage the decommissioning of legacy applications that are reaching end-of-life or are too costly to maintain, and support efforts to move applications from legacy infrastructure to cloud platforms.

Although this Guide takes an application-centric approach, it supports The Open Group vision of Boundaryless Information Flow™. Enterprises can gain better insight into the information needed to deliver business value by proactively governing their application landscape and improve the sharing of information across the enterprise.

Trademarks

Acknowledgements

The Open Group gratefully acknowledges the contribution of the following people in the development of this Guide:

Mats Gejnevall, Combitech

Mats Gejnevall is a long-time member of The Open Group and has been involved in many different standards and guides on architecture. He has been a co-chair of The Open Group SOA Work Group since 2006.

Mats works at Combitech as a mentor and architect helping organizations set up their Enterprise Architecture capabilities, creating architectures and roadmaps to transform their enterprises to adaptive future states. He has worked in all kinds of industries, from Government to Industry and from Telco to Defense.

Mats Lehrman, Capgemini

Mats Lehrman is an experienced Enterprise Architect at Capgemini. He is TOGAF® Certified and has been a strong supporter of the TOGAF standard since he first came into contact with it in early 2000.

Mats has knowledge and experience from a broad range of different architecture engagements at the enterprise, segment, and capability level, as well as projects with a focus on establishing and evolving Enterprise Architecture capabilities to increase the Enterprise Architecture maturity level at different organizations. He has delivered architecture engagements in both the public and private sectors during his long career.

Paul L. Williams, Capgemini

Paul Williams has been a member of The Open Group for several years and has participated in various architecture work streams and white papers. He is the co-chair of The Open Group Application Architecture Work Stream, and a member of The Open Group Certified Architect (Open CA) Work Group.

Paul works at Capgemini as an architect, with over 30 years of experience spanning public and private sector organizations to help clients realize their business objectives through the appropriate use of technology. He is TOGAF® and ArchiMate® Certified and a Capgemini University Qualified Facilitator, teaching and mentoring the TOGAF standard to colleagues and clients.

Referenced Documents

The following documents are referenced directly in this Guide or the concepts discussed in this Guide are derived from these works.

(Please note that the links below are good at the time of writing but cannot be guaranteed for the future.)

- Cloud Computing Explained, White Paper (W115), May 2011, published by The Open Group; refer to www.opengroup.org/library/w115
- ISO/IEC/IEEE 42010:2011: Systems and Software Engineering – Architecture Description; refer to: www.iso-architecture.org/ieee-1471
- Legacy Evolution to SOA, Guide (G122), April 2012, published by The Open Group; refer to: www.opengroup.org/library/g122
- Maximizing the Value of Cloud for Small-Medium Enterprises, Guide (G123), April 2012, published by The Open Group; refer to www.opengroup.org/library/g123
- Microservices Architecture, White Paper (W169), July 2016, published by The Open Group; refer to: www.opengroup.org/library/w169
- The TOGAF® Standard, Version 9.2, a standard of The Open Group (C182), published by The Open Group, April 2018; refer to: www.opengroup.org/library/c182

1 Introduction

1.1 The Need for Application Portfolio Consolidation

Application portfolio consolidation is a change to the application landscape with the intention of delivering business value and enabling business agility by optimizing the number and types of applications in use across the enterprise.

The enterprise application landscape can evolve in an uncontrolled way over time to include a mixture of applications based on commercial software (Commercial Off-The-Shelf (COTS) packages), open source software, and custom-developed software. Business-purchased and business-developed applications, directly bought or created by the business to augment and extend application functionality, also form part of the application landscape. Such "grey IT" is often constructed using office suites and tools, although most enterprises do not officially recognize such applications as being in their application portfolio.

Typically, an enterprise does not undertake application portfolio consolidation as an end in itself, but as part of a wider piece of architecture work to deliver business outcomes. The rationale for embarking on application landscape simplification can be varied but typically occurs as a result of:

- Mergers with, or the acquisition of, another organization
- Divestments and de-mergers from an organization
- Changes in the IT operating model from decentralized (diverse) to centralized (unified)
- Wide-ranging changes to IT budgets; e.g., do more with less
- A strategic review, either at business or IT level, to assess whether the application portfolio aligns with the business strategy
- Changes in IT strategy pursuing models like outsourcing; for example, using cloud or third-party solutions

The drivers and underlying issues most frequently associated with application portfolio consolidation are listed in Table 1 (in no specific order):

Table 1: Drivers for Application Portfolio Consolidation

Reference	Driver	Issue
AC.D01	Human Resource Costs	High cost of specialized skilled resources, and of training.
AC.D02	Vendor Costs	High cost of vendor licensing, vendor support.
AC.D03	Application Overlap	Multiple distinct applications with the same or similar functionality.

Reference	Driver	Issue
AC.D04	Application Duplication	Multiple instances of the same application, often in different versions.
AC.D05	Ageing Applications	Ageing applications and associated skills reaching their end-of-life.
AC.D06	Capital Expenditure (CapEx) → Operational Expenditure (OpEx)	Changing financing models to reduce cost.
AC.D07	Cutbacks in IT Budgets	IT operational cost reduction, thereby maximizing IT budgets more in strategic initiative.
AC.D08	Shadow IT/Grey IT	Applications bought or built directly by the business but not integrated into the overall application landscape.
AC.D09	Application Landscape Growth	Application landscape has grown with new applications being added without consideration of the overall landscape, often as a result of business activities (e.g., acquisitions, mergers, etc.). New applications may fill gaps within the landscape but are siloed and not integrated with existing applications.
AC.D10	Application Cost	The costs of maintaining and supporting an application throughout its life.

Application portfolio consolidation can have a strong cost focus with the aim to reduce the overall management, maintenance, and support costs for the application portfolio. This is directly reflected in the Resource Costs, Vendor Costs, Application Costs, CapEx to OpEx, and Cutbacks in IT Budgets drivers described in Table 1.

1.2 Benefits from Consolidating Application Portfolios

There are common benefits to an enterprise when simplifying and consolidating their application portfolio:

- Improved alignment of application functionality with business strategy and objectives
- Reduced Total Cost of Ownership (TCO) of the application portfolio, because of:
  - — Lower application maintenance costs
  - — Lower cost of associated technology and infrastructure
  - — Improved IT capacity utilization
  - — Lower vendor licensing costs for applications, technology, and infrastructure
  - — Lower vendor support costs
  - — Lower need for specialized application and technical skills
- Simplified application landscape that is easier to support and manage

- Optimized application portfolio with no major functional duplication or overlap
- Retirement of aged applications that have, or were near to, end-of-life
- Reduced integration complexity
- Reduction in the number of infrastructure platforms and technology stacks
- Improved system characteristics including Scalability, Performance, Stability, and other Non-Functional Requirements (NFRs)
- Support for improved, standardized, and automated processes across the IT organization

A better understanding of the Baseline Application Architecture and landscape is a useful by-product, as it shows how the applications in the portfolio support critical business activities and will be an enabler for any future architecture work.

1.3 Creating a Business Case for Application Portfolio Consolidation

A business case for undertaking consolidation is not always required by an enterprise, but where it is produced it will explain to business stakeholders the rationale and justification for doing the consolidation in the way articulated in the architecture. The business case will provide an assessment of the cost to undertake the consolidation and the estimated benefits that will be achieved, and the time period over which the benefits will be realized.

The business case will have an overview of the TCO for each application within scope so that stakeholders can see what is spent to deliver business capabilities and provide further rationale for consolidation. Costs detailed in the business case will provide migration costs, retirement costs, re-integration costs, and other relevant costs to achieve the stated savings and benefits.

Essentially, the business case demonstrates the business and functional value of undertaking the consolidation to key stakeholders such as the Program Board, Architecture Board, and the Chief Information Officer (CIO)/Chief Data Information Officer (CDIO), and empowers them to make informed decisions about consolidation.

A good set of Key Performance Indicators (KPIs) is valuable to give to stakeholders within a consolidation dashboard to show the progress of consolidation activities, and provides linkage back to the business case.

1.4 Approach to Application Portfolio Consolidation

The approach follows the TOGAF® Architecture Development Method (ADM) process cycle with a "baseline first, target second" style. The Baseline and Target Architectures will potentially contain details for all the architecture domains, but the primary focus will naturally be on the Application and Technology domains as these will be the most impacted, followed by the Business and Data domains.

The Baseline and Target Architectures will be created over a number of iterations of the ADM, and in particular, a number of iterations of Phase C. There will be information gathered from preceding phases (Phase A, Phase B), subsequent phases (Phase D, Phase E), and iterations of

Phase C for both data and applications. The phases and the steps within each phase are highly iterative and should not be considered a linear workflow.

The usual architectural deliverables from this approach are:

- A Baseline Architecture, with a focus on Baseline Application Architecture and Baseline Technology Architecture
- A Target Architecture, with a focus on Target Application Architecture and Target Technology Architecture
- An Architecture Roadmap encompassing:

  — Prioritized work packages

  — Transition Architectures

  — Constraints for implementing transformation

  — Transformation plan with planned transitions to deliver consolidation

The Architecture Roadmap comprises any Transition Architectures required to deliver the application transformation described by the Target Architecture. The Architecture Roadmap charts the journey from the baseline state to the target state and should be pragmatic in providing islands of stability in the application landscape.

2 Definitions

2.1 Abstraction Levels

Abstraction is an architectural modeling mechanism for splitting a problem area into smaller problem areas that are easier to model and therefore easier to solve. Abstraction levels are hierarchical in nature, moving from high-level models to more detailed-level models.

In this Guide, we introduce the following abstraction levels:

- Contextual – focused on understanding the context in which an enterprise operates and the context for which architecture work is planned and executed

  The contextual abstraction layer is about answering WHY an enterprise is undertaking architecture work, the scope of work, and the motivation in terms of goals, drivers, and objectives.

- Conceptual – centered on decomposing the requirements to understand the problem, and what is needed to address the problem, without unduly focusing on how the architecture will be realized

  The conceptual abstraction layer is about determining WHAT is necessary to realize the requirements and is usually modeled using service models (business service, information system service, technology service) that represent desired behavior.

- Logical – focused on identifying the kinds of business, application, and technology components needed to achieve the services identified in the conceptual layer

  The logical abstraction layer is about identifying HOW an architecture can be organized and structured, in an implementation-independent fashion. There will potentially be several ways to group services into logical components, based on principles and other grouping criteria, providing different logical solution alternatives.

- Physical – manages the allocation and implementation of physical components to meet the identified logical components

  The physical abstraction layer is about determining WITH WHAT physical components the logical layer components can be realized. There will potentially be many ways to use physical components to realize logical components, based on principles and other grouping criteria, providing different physical solution alternatives.

2.2 Configuration Management Database

A Configuration Management Database (CMDB) repository contains configuration items that relate to IT assets deployed and used by an enterprise. Configuration items typically cover hardware, software, networking, locations, documentation, and people.

2.3 Extract, Transform, Load

Processes and tooling to extract data from a data source or sources, and to transform the extracted data into the correct structures suitable for loading into the data target. Extract, Transform, Load (ETL) typically includes activities such as data weeding and data cleansing as part of transformation.

2.4 Grouping Criteria

Criteria (including Architecture Principles) that can be applied at different abstraction levels to group services and components to identify different architecture alternatives.

Examples of grouping criteria include:

- Affinity with information produced or consumed by services/components
- Affinity with business capabilities that use services/components
- Affinity with business services that are automated or partially automated by applications
- Minimizing interactions between the services
- Minimizing interactions between the components

2.5 IT Service Management

IT Service Management (ITSM) activities are concerned with the implementation and operation of IT services to meet business needs, supported by tools for handling incidents, service requests, problems, and changes.

2.6 Model Kinds

Refer to ISO/IEC/IEEE 42010:2011 (see Referenced Documents).

Model kinds capture conventions for a type of modeling. To adequately frame a set of concerns a viewpoint can use one or more model kinds. Each model is constructed in accordance with the conventions established by its model kind, typically defined as part of its governing viewpoint.

2.7 Recovery Point Objective

The Recovery Point Objective (RPO) is the targeted point in time to which data must be restored, following an outage (disaster or disruption), in order to resume processing transactions in order to avoid unacceptable consequences associated with a break in business continuity.

2.8 Recovery Time Objective

The targeted duration of time and a service level within which a business capability must be restored after an outage (disaster or disruption) in order to avoid unacceptable consequences

associated with a break in business continuity. The Recovery Time Objective (RTO) is often applied to the systems and applications supporting a business capability.

3 Architecture Vision

3.1 Partitioning

Application portfolio consolidation efforts can be partitioned to simplify the development of architectures and the implementation of changes to the application landscape.

3.1.1 Depth

From a depth dimension, all four architecture domains (Business, Data, Application, Technology) will be covered to enable the consolidation outcomes to be physically implemented, and the impact upon the enterprise understood by providing traceability from application-related changes back to business operations (capabilities, services, actors, roles), and to motivation (goals, drivers, objectives).

The depth of analysis and understanding within each architecture domain will vary, with more focus in the Data and Application domains and lesser focus on the Business and Technology domains.

3.1.2 Breadth

From a breadth dimension, a number of criteria based on application characteristics can be used. The following are commonly used characteristics:

- Business domain/business capability
- Criticality to enterprise
- Interoperability between applications
- Commonality of underlying technology and/or shared infrastructure
- Geographic location
- Or any combination of the above criteria

Analyzing applications by business domain and business capability provides a clear separation that follows the lines of business within an enterprise. For example, Sales, Human Resources, and Procurement business domains could be handled as separate areas. This approach makes it easier to identify affected application owners, and therefore makes the communication of changes simpler.

The criticality of applications can be measured in a number of ways based on diverse business measures, such as:

- Amount of revenue generated
- Amount of revenue protected

- Importance of safety to people, assets, and other resources
- Regulatory compliance and statutory reporting
- Security
- Any combination of the above measures

Business criticality determines how important an application or set of applications are to the enterprise to enable it to fulfill its mission. Criticality can be measured in a number of different ways, including:

- Evaluations made by application owners, business users, etc. collected via questionnaires
- Automatic measurements taken by discovery and monitoring tools to calculate how often an application is used
- Service-Level Agreements (SLAs) and other NFRs such as RTO and RPO
- Rate of change in business capability
- Disaster recovery/disaster tolerance configuration

When looking at SLAs it is often misleading to look at service availability as the only measure of business criticality. A better measure is the service support hours that underpin the application, alongside recoverability factors like the RTO and RPO.

A holistic view across criticality measures will give a consistent view on whether an application is, or set of applications are, important to the enterprise.

It is important for consistency that an enterprise applies the same set of criticality measures to all applications being considered. An agreed scoring method that can be reliably applied by potentially different people at different times and uniformly interpreted will support this consistency. It is unlikely that every enterprise will measure criticality in the same way.

Another important application characteristic to consider when looking at the breadth dimension is interoperability between applications. Tightly-coupled applications will need to be considered together to prevent destabilizing the applications and the business services they support. For example, applications that are integrated on a point-to-point basis create dependencies, meaning the applications need to be considered as a discrete set.

This consideration also applies to applications that use the same underlying technology, or are deployed to shared infrastructure, or deployed to the cloud. Whilst there may not be a hard dependency between applications that have the same technology in common, or share infrastructure, there can be economies of scale in considering them as a related group.

3.1.3 Time

From the perspective of time, there are different time-related criteria that may influence how the architecture is partitioned, including:

- The longevity of an application’s business usefulness
- The time before an application reaches its end-of-life/support from their vendor

- Business-critical periods such as peak processing periods, seasonal events, and regulatory/statutory events
- Budgetary periods and financial year-ends

3.2 Stakeholders and Stakeholder Concerns

Stakeholders who typically have concerns and interests in application portfolio consolidation range from the CIO/CDIO and Architecture Board to Application Owners and the various business users who use the applications (see Table 5 for a non-exhaustive list of example stakeholders).

Note that:

- Application Architects and Enterprise Architects are responsible for producing the Application Architecture for consolidation
- The Architecture Board and Program Board are accountable, as they will govern the architecture and implementation, respectively
- Various stakeholders will be consulted for their perspectives and impacts, including Corporate Security, Application Management, Application Owners, Data Guardians/Stewards, Infrastructure Management, Technology Architects, and Security Architects
- Other stakeholders will be kept informed of progress, including the CIO/CDIO, Business Users, Procurement, Service Management, Program Management Office (PMO), and Human Resources
- Subject Matter Experts (SMEs) are responsible for evaluating the technical and functional fit of the application and producing the recommendations for consolidation, including decommissioning, replacement, and application changes

The Architecture Board has overall governance of the consolidation architecture and implementation. However, the CIO/CDIO is an important stakeholder for consolidation as they are concerned with both driving enablement and reducing costs. The CIO/CDIO is most likely to be consulted on consolidation impacts in delivering enablement.

Stakeholders will have a number of concerns about application portfolio consolidation including knowing what applications are in the landscape, what business capabilities rely on specific applications, and what is the cost to run and maintain those applications (see Table 6 for an example catalog of common stakeholder concerns).

The views needed to address stakeholders' concerns will be created iteratively as the architecture effort gathers information and progresses through the different architecture levels (conceptual, logical, physical) to build up a coherent picture of the architecture.

In many cases, a core team is created and series of workshops conducted with the appropriate participation of the stakeholders. The workshops help with:

- Initial information gathering
- Validation and verification of findings and recommendations

- Continuing stakeholder participation and transparency

4 Elaborating the Baseline Application Architecture

4.1 Overview

When deriving the Baseline Application Architecture, consider what resources are available for reuse from the Architecture Repository, but validate the information and assess whether the Baseline Architecture Building Blocks continue to reflect the baseline state.

Understanding the Baseline Application Architecture starts at the physical abstraction layer in a bottom-up way:

- Physical abstraction layer:
  - — Document the applications in scope
  - — Document application structure and behavior
  - — Document application interactions
  - — Document what data is produced and consumed by applications
  - — Document what technologies are used by applications
- Logical abstraction layer:
  - — Document the logical components that the applications and technologies realize
  - — Document logical component interactions
  - — Document the mapping of data produced and consumed by applications to logical components
- Conceptual abstraction layer:
  - — Document the services that the logical components realize
  - — Document service interactions, including service qualities (derived from the physical and logical abstraction layers)
  - — Document the mapping of information produced/consumed by logical components to services
  - — Document the mapping of services to business activities

A useful technique in understanding the baseline application landscape is the T-Analysis approach (see Appendix D), where the analysis can be split into several passes with decreasing breadth and increasing depth, with each successive pass providing more detail about the baseline state.

4.2 Physical

Activities at the physical abstraction layer include making an inventory of the applications in scope, and then for each application gathering details about its behavior, identifying information produced and consumed by it, determining the technologies it uses, and collecting details about its running, maintenance costs, and service qualities.

The baseline viewpoints for the physical abstraction layer primarily lead to collecting information about physical application components, physical data components, and physical technology components. See Appendix C for example model kinds, viewpoints, and views.

A prerequisite to understanding what opportunities exist for consolidation is making an inventory of the applications in use by the enterprise, the information produced and consumed, and the technologies that support the operation of those applications. Therefore, it is important to gain a complete perspective of all the different variants of applications and technologies in the landscape. This means cataloging each vendor version of an application and includes any "in-house" custom developed applications.

Sources of information about the physical landscape can include CMDB, ITSM, vendor product documentation, specifications for "in-house" custom developed applications, and documentation from partners who provide third party applications.

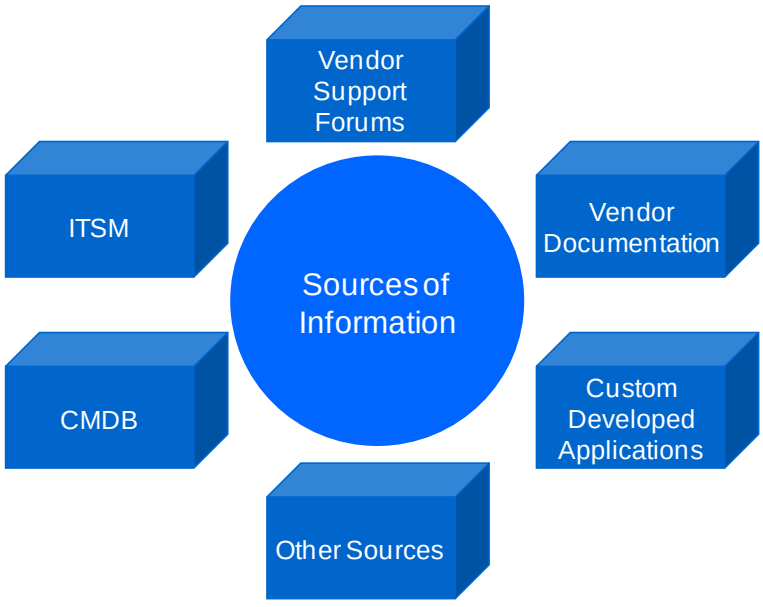

Figure 1: Sources of Information

This is not an exhaustive list and other sources of information may be available within organizations.

The physical databases, repositories, and other types of data stores accessed by applications are sources of information about which applications act as systems of record, or master certain types of information within the enterprise. The analysis will also provide insight into the complexity of any data transformations that occur when exchanging information between data stores and applications.

Assessing security aspects at the physical abstraction level includes understanding the mechanisms employed for user authentication and authorization, the mechanisms for auditing, and the use of any security enforcement products, such as asset protection, that may be within scope.

The following measures can be determined from the physical inventory of the baseline (see Appendix B):

- The service levels and other non-functional aspects for each application (measure AC.MS22)
- Who supports the applications (measure AC.MS09)
- Which team(s) develop and maintain "in-house" custom-developed applications (measure AC.MS09)
- Who are the users of each application, and who is the designated business owner (measure AC.MS01)
- Whether an application provides regulatory or statutory reporting functionality (measure AC.MS03)
- What are the costs for each application:
  - — Maintenance costs (measures AC.MS04, AC.MS05)
  - — Vendor licenses (measure AC.MS06)
  - — Run and support costs (measure AC.MS07)
  - — Infrastructure costs (measure AC.MS08)

Use the physical inventory information gathered from available ITSM systems to determine:

- The number of users for each application (measure AC.MS02)
- The volume of change requests made over the last year per application (measure AC.MS10)
- The number of incidents reported per application (measure AC.MS20)
- The amount of unplanned downtime during the last year per application (measure AC.MS21)
- The number of SLA breaches per application (measure AC.MS22)

These measures will indicate the volatility and stability of each application. Further measures, such as Skill Paucity (AC.MS14) and Source Code Unavailability Risk (AC.MS11), etc. are difficult to quantify and so are better measured using a heat map.

It is important to recognize that information about the physical level should be properly maintained within the Architecture Repository and asset management in order to accurately determine the above measures. Where such knowledge becomes stale and out-of-date, a physical inventory will need to be taken to refresh the information.

4.3 Logical

Activities at the logical abstraction level include identifying the logical application, data, and technology components in scope, deriving the interactions between the components, and aggregating maintenance and run costs at a logical component level.

The baseline views for the logical abstraction layer are concentrated on gathering details about logical application components, logical data components, and logical technology components. See Appendix C for example model kinds, viewpoints, and views.

The focus is a lightweight iteration to establish, to sufficient granularity, the kinds of applications that are being used by the enterprise and how these map to business activities. The effort expended should be adequate to collect the measures needed to support decision-making when working on the Target Architecture.

A review of the Architecture Repository should be done first to identify existing resources that describe logical application, data, and technology components. Where this information is missing or incomplete, then effort is required to abstract this information from the physical inventories taken earlier in the iteration. For example, logical application components can be derived from vendor documentation for packaged software, and from specifications for custom-developed applications.

As logical application components abstract the detail of physical application components and provide an implementation-independent representation, they provide an inventory of the kinds of applications that an enterprise has. This has the benefit of making it easier to identify application overlap and duplication; for example, where two commercial packages are abstracted at a logical level as being Customer Relationship Management (CRM) applications, thus highlighting a potential duplication. It may be necessary to decompose logical application components to a lower level of granularity, particularly for a packaged application, to identify if there is actual duplication and overlap with other applications. However, it should be emphasized that the modeling effort undertaken should be proportionate.

It is useful to use an application taxonomy to categorize the kinds of applications that an enterprise has. It is important to establish a taxonomy that fits the enterprise; i.e., a transport company will have a different taxonomy to a government department, although there will probably be some common elements. It is considered good practice to use a taxonomy that is aligned with the business domains within an enterprise, as one of the benefits is better governance of application proliferation and associated costs. Another benefit is enabling easier traceability from application to business service.

Deriving logical data components from the physical data components supports analysis on the types of data stores in the landscape and aids identification of where data may be siloed, how data is shared and referenced, and where data is potentially duplicated if it is mastered in several places.

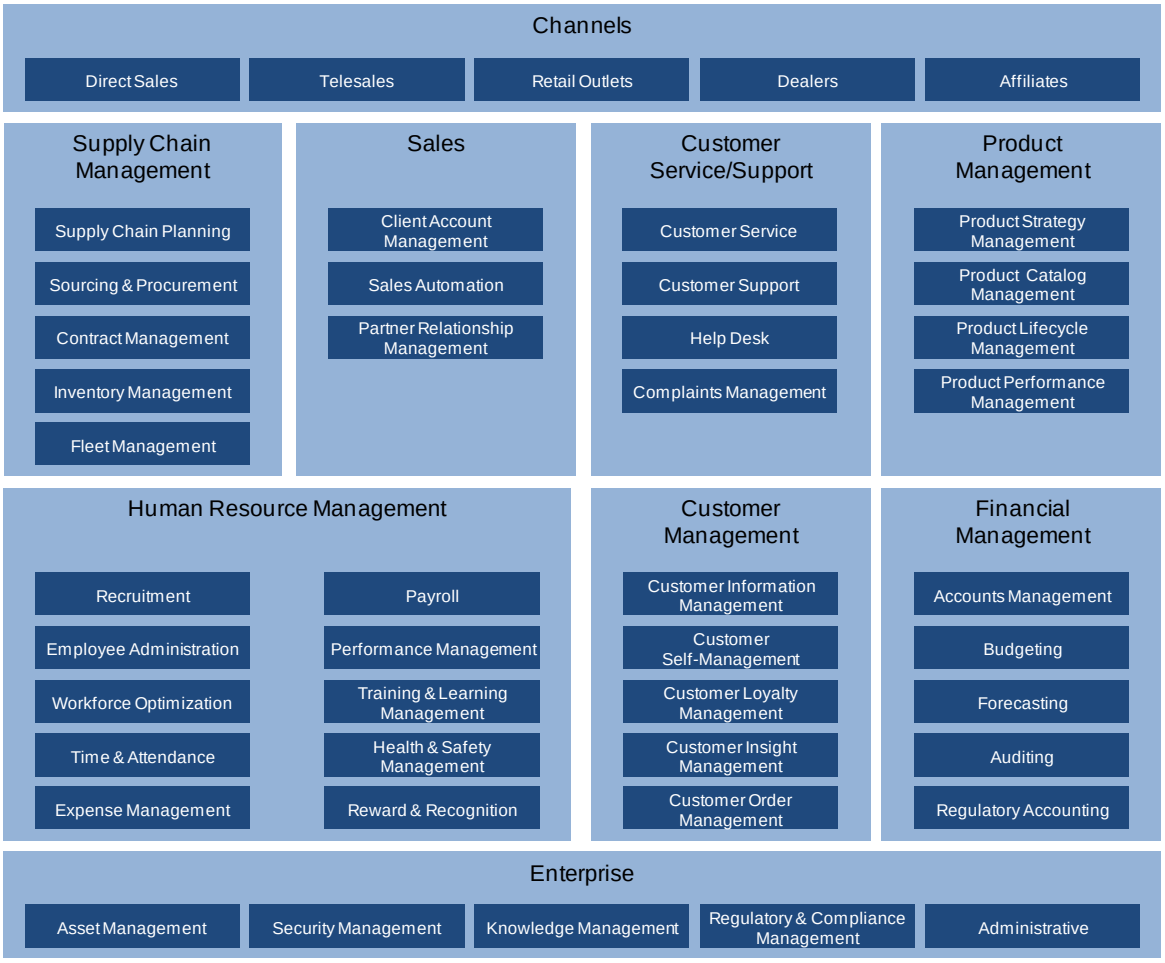

Figure 2: Application Taxonomy Example

The analysis also supports creation of information security models showing which roles have permission to access data.

The following measures can be determined for each logical component, derived by combining and aggregating captured physical component measures (see Appendix B):

- Whether a logical component supports business-critical activity (measure AC.MS01)
- The total number of users for each logical component (measure AC.MS02)
- Whether a logical component supports regulatory or statutory reporting functionality (measure AC.MS03)
- The costs for each logical component:

  — Maintenance costs (measures AC.MS04, AC.MS05)

  — Vendor licenses (measure AC.MS06)

  — Run and support costs (measure AC.MS07)

  — Infrastructure costs (measure AC.MS08)

Other measures are typically more difficult to aggregate at a logical level or can often provide a misleading perspective.

Measures are extrapolated from the physical level and aggregated at a logical level to provide an assessment of the logical application components health, value to the business, number of users, and run costs.

4.4 Conceptual

Activities at the conceptual abstraction layer include identifying information system services that support business services, gathering details about the service qualities, contracts, and measures for each identified information system service, and supporting the technology service, respectively.

The baseline views at the conceptual abstraction layer include capturing information about information system services, data entities, business services, processes, capabilities, and technology services. See Appendix C for example model kinds, viewpoints, and views.

When collecting information about services (information system service, business service, technology service) details about service qualities and contracts between services also need to be captured to gain a fuller picture of traceability between business and applications, and interdependencies between applications and technology. Understanding the service levels, defined within contracts, allows proper impact analysis to be done.

Discovery and analysis efforts at the conceptual abstraction layer are the final piece in understanding what business activities are supported by applications and technology. This provides the traceability from applications to business operations and activities.

4.5 Summary

The Baseline Architecture conveys a picture of what applications are within the portfolio, how they are structured, how they behave and cooperate with each other, what information they produce and consume, how they are secured, what technologies they use, and how much it costs to run and maintain them.

The Baseline Architecture provides a holistic view of the applications within the landscape at different levels of abstraction suitable for communicating with a diverse range of stakeholders and is the basis for developing the Target Architecture and identifying opportunities to optimize the application portfolio to better deliver business outcomes.

5 Defining the Target Application Architecture

5.1 Overview

The high-level steps in defining the Target Application Architecture are:

1. Establish the consolidation actions for each application within scope – for example, whether to retire an application, replace an application, or some other remedial action
2. Determine the timeframe within which the consolidation actions are to be enacted
3. Develop the Architecture Roadmap and associated plan for delivering the identified consolidation actions within the defined timeframes

To recap, the TOGAF approach to determining the Architecture Roadmap is a gap analysis between the Baseline Architecture and the Target Architecture.

Developing the Target Application Architecture will take several iterations, depending on the actual drivers behind the need for consolidation and the principles defined by the enterprise. Principles and other grouping criteria will be applied at conceptual, logical, and physical levels to provide alternative Target Application Architectures, offering choices in how the consolidation can be achieved.

Defining the Target Application Architecture starts at the physical level, by taking the information gathered from the Baseline Architecture and determining where to focus efforts to achieve best results. The focus of developing the Target Application Architecture revolves around clearly identifying the consolidation actions needed for each application, and the timeframe required to complete those actions and achieve the expected business benefits.

One approach that works well in practice is to group applications into a number of distinct time periods, and the 3Ts – as defined in Table 2 – is such an approach, where applications are segmented into three time periods.

Table 2: The 3Ts Grouping Approach

Term	Description
Immediate-term (Do Now)	Applications that can be changed in a short time period; for example, within 3 to 6 months.
Near-term (Do Next)	Applications that require a longer time period to amend; for example, within 6 to 24 months.
Long-term (Do Later)	Applications that either require a much longer time period to change or where there is some doubt as to what action to take.

To help determine the consolidation actions needed for each application, a classification scheme can be used alongside the 3Ts using some uniformly applied evaluation criteria. The 5Rs – as

defined in Table 3 – is such a classification scheme, and consolidation actions are determined based on an application's perceived health and business value.

Table 3: The 5Rs Classification Scheme

Classification	Description
Retire	De-commission the application and associated data, and remove from the portfolio.
Replace	Replace and migrate the application and data to another application, which may be an existing application or a new application in the portfolio; and may include replacing the application with a Software-as-a-Service (SaaS) offering.
Re-platform	Move the application to a different platform; e.g., move from physical infrastructure to virtualized infrastructure, or to the cloud – Platform-as-a-Service (PaaS) or Infrastructure-as-a-Service (IaaS).
Re-architect	Re-design the application to improve its health; e.g., upgrade to a more resilient, more modern database; potentially may also include re-platforming.
Refactor	Enhance and improve the application.

To help stakeholders understand the 5Rs scheme it is helpful to visualize the scheme as a grid, as shown in Figure 3.

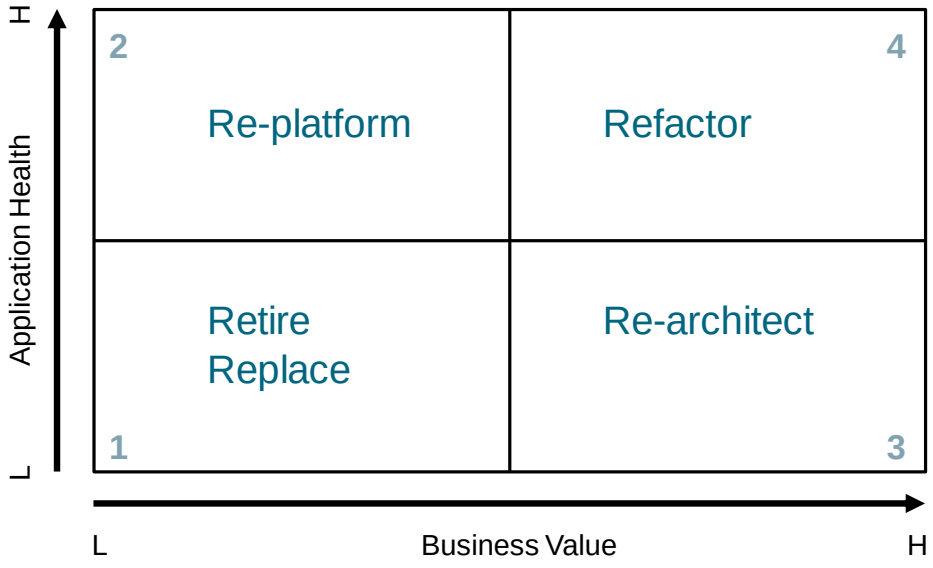

Figure 3: 5Rs Visualization

The vertical axis indicates an application's health and the horizontal axis indicates an application's business value. A measurement of an application's health will consist of several measures such as Number of Change Requests Last Year (AC.MS10), Source Code Unavailability Risk (AC.MS11), Documentation Unavailability Risk (AC.MS12), Number of

Incidents Last Year (AC.MS20), and other such measures (see Appendix B). Similarly, an application's value can be measured based on Business Criticality (AC.MS01), Regulatory Need (AC.MS03), the amount of revenue it generates or protects, and so forth.

The state of an application's health and the value it provides to the enterprise will determine the application's position on the grid:

- Quadrant 1 (low health, low value) – either retire or replace the application
- Quadrant 2 (high health, low value) – re-platform the application
- Quadrant 3 (low health, high value) – re-architect the application
- Quadrant 4 (high health, high value) – refactor the application

Using the 5Rs enables applications to be grouped into time periods as defined in the 3Ts. This mapping will depend on other factors within an enterprise such as business priorities and strategic initiatives that are already in-flight, and so will likely vary between different enterprises. This is further discussed in Section 5.9.

Other visualizations can be used to explain to stakeholders the state of applications in the portfolio. For example, Figure 4 shows the relative running costs of a group of applications against their value to the enterprise. Additional information, such as the application's health, can be visualized as colored circles using a traffic lights scheme (red-amber-green).

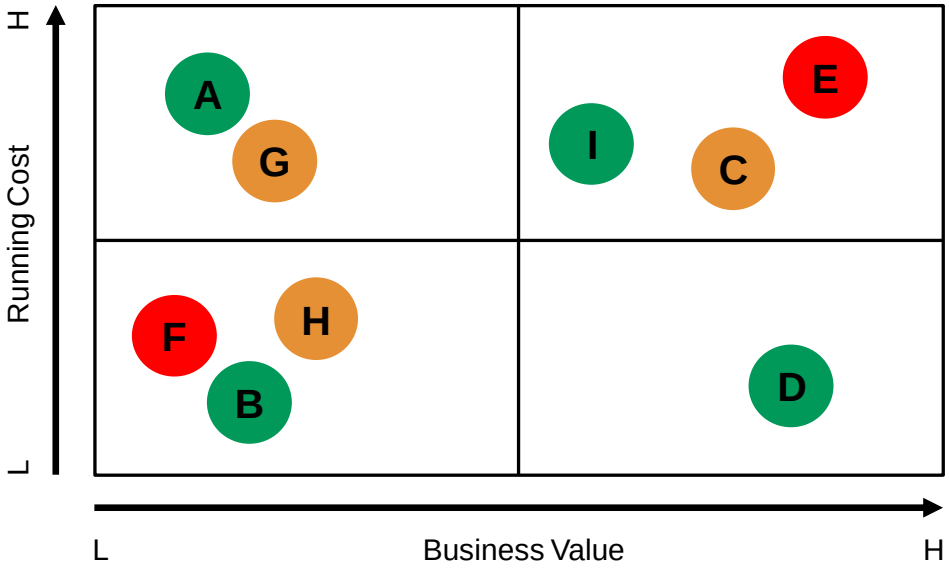

Figure 4: Cost *versus* Value Visualization

5.2 Levels and Iterations

Developing the Target Application Architecture will happen over many iterations and involve switching between the different levels of abstraction to ensure any impacts on business capabilities and services are clearly identified and understood.

Switching between the abstraction levels typically follows the following pattern:

- Physical level – determining the changes required to physical application and technology components for the specific consolidation action (e.g., retire, replace, etc.)
- Logical level – abstracting the physical component changes into logical component changes
- Conceptual level – mapping the logical component changes onto business capabilities and services to establish the impact
- Repeating the above for each physical component change until all models are consistent, and all impacts understood

Figure 5 illustrates the switching between the abstraction levels, and the iteration of modeling efforts:

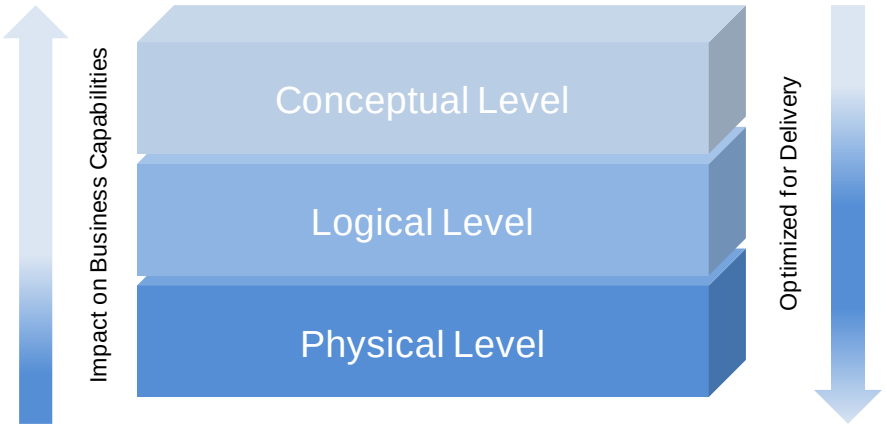

Figure 5: Switching Between Abstraction Levels

When traversing between each abstraction level, use the T-analysis approach and gap analysis techniques to determine if the right level of granularity has been achieved, and to help focus modeling efforts, and control iteration.

Consider the influence of Architecture Principles, and other grouping criteria (such as affinity with information, affinity with roles, etc.), on the applications within each group to identify Target Application Architecture alternatives. Other criteria such as security, governance, data migration, information mastering, and information sharing will also influence Application Architecture alternatives.

Each alternative will have distinct advantages or disadvantages that will need to be discussed and agreed with stakeholders. Additional viewpoints and views may be needed to allow stakeholders to explore the alternatives and understand any dependencies, risks, and uncertainties.

Use trade-off analysis to resolve conflicts between alternative Application Architectures, focusing on, but not limited to:

- Stakeholder requirements and concerns
- Time and cost of realizing each alternative, including any transitions and plateaus ("islands of stability")

- Impact on business capabilities during implementation of each alternative
- Time period over which estimated benefits of each alternative will be achieved
- Risks associated with each alternative and any mitigating actions needed

Use viewpoints AC.VP09 and AC.VP11 (see Appendix C) to generate views for stakeholders showing the alternative Application Architectures and trade-off analysis undertaken, and how applications are grouped for consolidation.

5.2.1 Physical Level Considerations

The majority of the work in defining the Target Application Architecture will happen at the physical level.

The focus is to create the Target Application Architecture, or to identify alternative Target Application Architectures where the Architecture Principles in use by the enterprise lead to more than one solution. This will include adding any new physical components needed, removing any redundant physical components no longer required, and updating the usage of any remaining physical components throughout the various models and views.

It is important to understand the dependency between applications, particularly if an application within the scope of consolidation affects an application that is outside of that scope (and which may not have been considered or reviewed in detail). This is a critical activity for applications classified as retire, replace, and re-platform.

Interoperability and how applications interact with each other is an important aspect. However, dependencies between applications are not limited to interoperability or the exchange of data. Other aspects include, for example, whether an application masters specific data and the potential effect on business capabilities if that application is replaced and another application becomes master for the data.

Another aspect to consider is the impact on the NFRs, or service qualities, of applications that are being re-platformed, refactored, or re-architected, and whether such service qualities will change as a consequence of the consolidation work. For example, what is the impact on the RTO and RPO of the applications? This consideration equally applies to replacement applications, to ensure such substitutions have sufficient capacity to take over the workloads of the applications that they are supplanting.

Other considerations at the physical level focus on the approach and tools for performing any data and application migration activities, whether the migration will cross network boundaries, and how such activities will be governed; i.e., what security mechanisms will be needed to ensure integrity and confidentiality of the information and application being migrated.

The following viewpoints will be populated with physical data, application, and technology components to support conversations with stakeholders:

- AC.VP02 Application Structure
- AC.VP03 Application Behavior
- AC.VP04 Application Information
- AC.VP05 Application Cooperation

- AC.VP07 Technology Cooperation

There may be distinct views generated, from the above viewpoints, for each alternative Target Application Architecture. The purpose of generating the views is to ensure consistency at the physical level, and to enable traceability back to business services and capabilities.

5.2.2 Logical Level Considerations

An important part of modeling at the logical level, when consolidating the application portfolio, is that it acts as a conduit to the conceptual level where business operations and capabilities are considered. The focus is a lightweight iteration to establish that traceability between the physical and conceptual levels.

Mapping physical components to logical components also provides a consistency check and a mechanism for determining the impact to common components that may be utilized and shared with other applications that are not part of the consolidation scope.

The following viewpoints will be populated with logical data, application, and technology components:

- AC.VP02 Application Structure
- AC.VP04 Application Information
- AC.VP07 Technology Cooperation

There may be distinct views generated for each alternative Target Application Architecture identified at the physical level.

5.2.3 Conceptual Level Considerations

The focus at the conceptual level is the traceability of changes required at the physical level to business services, to determine that the effects on the business capabilities and services are clearly understood and are intended. The aim is to enable business continuity planning during the consolidation work, with potential disruption to business operations identified early.

The following viewpoints will be populated with logical data, application, and technology components mapped to the business services and capabilities:

- AC.VP02 Application Structure
- AC.VP03 Application Behavior
- AC.VP04 Application Information

There may be distinct views generated for each alternative Target Application Architecture identified at the physical level.

Additional views that provide traceability to, and impact on, business services and business capabilities can be used.

5.3 Target Application Architecture Checklist

The checklist given in Table 4 can be used as an *aide memoire* and to help shape the thinking about what needs to be considered when consolidating an application, or group of applications.

Table 4: Target Application Architecture Checklist

Reference	Item	Description
AC.CK01	Data Change	What change to data is required? This considers data structural changes and the location of data (including archive locations); and any data migration, cleansing, and transformation of data needed.
AC.CK02	Data Access Change	What data access change is required? This considers authentication, authorization, auditing, and asset protection.
AC.CK03	Application Change	What application change is required? This considers application structural changes, and transitioning to different application components.
AC.CK04	Application Access Change	What change to application access is required? This considers authentication, authorization, and auditing at an application level.
AC.CK05	Technology Change	What change to technology is required? This considers technology transitions and migrations required to deliver consolidated applications.
AC.CK06	Infrastructure Change	What infrastructure change is required? This considers changes needed to infrastructure (servers, network, storage, etc.) to deliver consolidated applications.
AC.CK07	Business Continuity	How is business operational continuity maintained? This considers how continuity is managed, and keeping disruptions to business capabilities to a minimum.
AC.CK08	Process Change	What business process change is required? This considers changes needed to business processes as a result of application consolidation.
AC.CK09	Organizational Change	What change to organizational structure is required? This considers changes need to the organizational structure as a result of application consolidation, and includes IT and operational business units.
AC.CK10	Training	What training is required, and for whom? This considers the training needs of technical staff and business users

5.4 Retiring an Application

The decision to retire or replace an application occurs because it is assessed as having low application health and offering low business value; i.e., it appears in the lower-left quadrant in the 5Rs visualization diagram (Figure 3). Deciding the correct course of action – retire or replace – will depend on one or more of the measures listed in Appendix B.

Activities typical for retiring an application are to:

- Identify application and infrastructure services needed to facilitate the retirement of an application; this includes the need for archiving of data, migration of data, movement of information system services; i.e., to be included into another application component
- Update the Target Architecture to include any new services and components needed
- Remove from the Target Architecture those information system services and components related to applications being retired
- Update interactions between services and between components based on the above activities
- Generate or update views describing application cooperation, application portfolio, application structure, application behavior, application information, technology portfolio, and technology usage

The main consideration for retiring an application is what happens with the associated data that the application masters or provides as reference data to other applications in the portfolio (see checklist item AC.CK01 Data Change in Table 4).

The following options are usually the main choices for retaining the data:

- Archive the data to offline storage and restore when required; useful if the data is infrequently needed
- Archive the data to online or nearline storage, such as a data mart, and query when needed; useful when the data is frequently required but only for reference; i.e., the data will not be updated
- Transform and migrate the data to another application within the portfolio; useful if the data has business value and will be updated

Each of these options will affect how the data will be accessed (see checklist item AC.CK02 Data Access Change in Table 4) after the owning application is retired. Consider creating a special data enclave if retiring many applications where the data is still required for reference purposes.

Where a central authentication and authorization capability is utilized for application access, a housekeeping clean-up task will need to remove entries specific to the retiring application to prevent potential security loopholes (see checklist item AC.CK04 Application Access Change in Table 4).

Also consider what impact retiring an application has on the technology and infrastructure it uses (see checklist items AC.CK05 Technology Change and AC.CK06 Infrastructure Change in

Table 4). For example, can the infrastructure also be retired, or a reduction achieved, thus further reducing the operational cost, or the infrastructure reused thus saving capital cost.

Retiring an application will impact the business processes operated by an enterprise, and potentially the organizational structure of that enterprise (see checklist items AC.CK08 Process Change and AC.CK09 Organizational Change in Table 4). For example, by retiring an application there may be an opportunity to retire some or all of the business services (and the channels in which the services are offered) that the application supports, and as a result restructure the business units to achieve operational cost savings by reducing the Full-Time Equivalents (FTEs) within the business units.

For an application that uses specialized technology or infrastructure which is no longer needed after consolidation, operational cost savings could be realized by the IT organization through a combination of a decrease in licensing and support costs, and potentially FTE cost reduction if personnel with such specialized knowledge are not retained by the enterprise.

5.5 Replacing an Application

The decision to retire or replace an application occurs because it is assessed as having low application health and offering low business value; i.e., it appears in the lower-left quadrant in the 5Rs visualization diagram (Figure 3). The decision to replace an application is usually made because the application is either healthier or of more business value than applications classified as retire.

Activities typical for replacing an application are to:

- Identify information system services related to the application marked for replacement
- Create a set of solution alternatives based upon Architecture Principles
- Choose one solution alternative and populate with new physical components
- Update the Target Architecture to include new or updated logical and physical components
- Remove from the Target Architecture those application components related to applications being replaced
- Update interactions between services and between components based on above activities
- Generate or update views describing application cooperation, application portfolio, application structure, application behavior, application information, technology portfolio, and technology usage

The primary considerations for replacing an application are how to transition business users and supported business capabilities to the target application while maintaining normal business operations. These considerations include how to migrate any data from the source application's data store(s), how to transfer or replicate needed functionality to the target application, how to migrate users to the target application, and the impact on the target application's current infrastructure and service qualities.

Migrating data from the source application to the target application (see checklist item AC.CK01 Data Change in Table 4) may involve the wholesale ingestion of data, or partial ingestion of

some of the data with other data being archived. This often requires a change to the internal data structures of the target application's data store(s), with data from the source application needing to be weeded, cleansed, and translated to the correct data structures. Such translation can be done by scripts, database stored procedures, and so on, but for large or complex translations an ETL tool is frequently necessary.

Migrating data will impact how the transitioned data is accessed in the target application (see checklist item AC.CK02 Data Access Change in Table 4). This largely depends on what data is being migrated, whether it is commercially-sensitive or contains information relating to individuals, the regulatory framework the enterprise operates within, and how the data is migrated to the target application.

Combining information from the source application with data already held in the target application may change the way the data is classified, resulting in additional security controls being needed to safeguard the data, and impacting the way the data is audited or information assets protected. Data guardians/stewards and corporate security will be principal stakeholders for this aspect of data.

There may be a need to replicate some of the functionality from the source application to the target application (see checklist item AC.CK03 Application Change in Table 4) in support of migrated data, business processes that will be retained but supported by the target application, or for interoperability with other applications. This could involve purchasing additional modules for a COTS package or undertaking software development.

Moving some or all of the users from the source application to the target application will involve the cleansing of redundant, unused, or inactive accounts; e.g., those that have not been used within the last 12 months (see checklist item AC.CK04 Application Access Change in Table 4). User accounts within the target application will need to be set up and activated, and the associated users on-boarded and provided with suitable training on how to use the target application (see checklist item AC.CK10 Training in Table 4).

Depending on the size of the user community being transferred to the target application, this activity may be a simple exercise or may be split into groups of users based on selected criteria; e.g., geographic area, business unit, user type, etc.

Technology considerations (see checklist item AC.CK05 Technology Change in Table 4) and infrastructure considerations (see checklist item AC.CK06 Infrastructure Change in Table 4) are equally focused on the source and target applications. As part of transitioning to the target application, technology upgrades may be required, and additional infrastructure capacity provisioned to meet any increase in workloads and maintain the service levels of the target application. Any potential reductions or retirement of technology and infrastructure will be the same considerations as for retiring an application.

Maintaining normal business operations during the transition from source application to target application will depend upon several internal and external factors (see checklist item AC.CK07 Business Continuity in Table 4). Conducting a Business Transformation Readiness Assessment (see the TOGAF standard in Referenced Documents) will identify factors beyond typically elements like regulatory conformance and peak business periods.

For the source application being replaced, process and organizational considerations (see checklist items AC.CK08 Process Change and AC.CK09 Organizational Change in Table 4) will be the same as for retiring an application. There will be opportunities to realize cost savings

through retiring or reducing the number of business processes operated, and therefore the number of FTEs required.

A major factor in the successful replacement of an application is repeated rehearsal with the business. Rehearsals should involve trial runs of key migration activities such as data migration, user migration, and application access. Any significant issues detected during rehearsals should be addressed ahead of the actual switch over.

5.6 Re-Platforming an Application

An application classified as re-platform has high health but relatively low business value; i.e., it appears in the top-left quadrant in the 5Rs visualization diagram (Figure 3). This means the applications are still needed by the enterprise but are less critical to the operation of the enterprise than other applications. The main difference between re-platforming and re-architecting is that applications that need re-architecting are classified as having low health and therefore need more work to rectify than moving to a different platform.

Consider the influence of Architecture Principles to distinguish alternate ways of re-platforming, such as moving to shared infrastructure, utilizing on-premise virtualization and containerization, or moving to hosting in the cloud. Other criteria, such as the measures listed in Appendix B, can be leveraged to help identify components that may need to change during the design of the Target Architecture.

Activities typical for re-platforming an application are to:

- Identify any components to be replaced based on measures such as cost (AC.MEAS.08), platform stability (AC.MEAS.23), and platform lifecycle (AC.MEAS.19)
- Include constraints set by principles, sourcing strategy, corporate standards, and compliance requirements to set the game plan for re-platforming
- Create a set of possible solution alternatives based upon Architecture Principles
- Choose one solution alternative
- Populate and update the Target Architecture to include new components and services
- Remove from the Target Architecture those technology services and components that are being replaced
- Generate or update views describing application cooperation, application portfolio, application structure, application behavior, application information, technology portfolio, and technology usage

The main considerations for re-platforming an application are technology changes, infrastructure changes, and business continuity (see checklist items AC.CK05 Technology Change, AC.CK06 Infrastructure Change, and AC.CK07 Business Continuity in Table 4). Other considerations that may apply are where the data and application are relocated to and therefore accessed from (see checklist items AC.CK02 Data Access and AC.CK04 Application Access in Table 4), but this will depend on the kind of re-platforming done.

Re-platforming does not usually affect business processes, organizational structure, or require re-training of business personnel (see checklist items AC.CK08 Process Change, AC.CK09

Organizational Change, and AC.CK10 Training in Table 4). However, training may be required for IT personnel if the re-platforming uses a completely different set of technologies and services.

Any potential reductions or retirement of technology and infrastructure for the old platform will have the same considerations as for retiring an application. Similarly, maintaining normal business operations during re-platforming include the same considerations as for replacing an application, and like replacing an application, a major factor in successful re-platforming is repeated rehearsal with the business.

5.6.1 Shared Infrastructure

When moving an application to on-premise shared infrastructure (whether physical or virtualized), capacity planning is an essential activity to ensure all likely residents of the infrastructure can be accommodated, and capacity grown if and when needed. The design activities undertaken will depend on whether the shared infrastructure already exists or is being created as part of the consolidation work. The shared infrastructure may use a different set of technologies than the applications being moved to it; for example, to improve service qualities such as availability and recoverability. These changes of technologies should, in most cases, be transparent to the applications, or require minor modifications.

Other considerations for shared physical infrastructure are the technologies needed for migration of applications and data onto the shared platform, how access is provided, and what tooling is needed for management of the platform. These considerations may bring new technologies and services into the architecture, new management processes, and potentially new training for IT personnel.

5.6.2 Virtualization

Migrating an application to an on-premise virtualization platform has many aspects in common with the shared physical infrastructure approach, but with the main difference being that a virtualization platform can support a heterogeneous set of operating systems and environments. Although there is an underlying physical infrastructure on which the virtualization platform resides, from an application perspective, the use of physical machines is replaced by virtual machines.

However, capacity planning remains an important IT activity for on-premise virtualization. Prior to moving an application to a virtualization platform, tools for assessing the current configuration and workload of the application on physical machines will be required to accurately size the virtualization platform, and to give assurance that the workload can be accommodated without compromising other applications that are, or will be, resident.

Additionally, specialized tools will be needed, which may be vendor-specific, to clone or migrate an application from the physical machines it uses to the virtualization platform. To make the most effective use of tools, IT personnel will need training on any new tools introduced to operate new processes when managing the virtualization platform.

Alternatively, approaches to infrastructure provisioning, such as automation, DevOps, and Infrastructure as Code (IaC) may introduce a distinct set of new tools and associated new training requirements for IT personnel.

There is one aspect of migrating an application to a virtualized environment that is often overlooked, and that is the impact of licensing – licensing of the application itself (if a COTS product) and of the technologies it uses. Some vendors may have restrictive or prohibitively expensive licensing policies for virtualized platforms; e.g., you have to buy a license based on the underlying physical infrastructure used for the platform rather than the actual number of virtual machines on which the application or technology runs. This kind of constraint should be clearly understood up-front when assessing whether to move an application to a virtualized platform.

5.6.3 Cloud

Cloud hosting is a wide-ranging topic and not all aspects are covered in this Guide. Aspects such as assessing cloud providers, choosing a preferred cloud provider, procuring cloud services, and engaging in commercial negotiations are outside the scope of this Guide. The Open Group has produced several documents focused on cloud and cloud platforms, which can be found in The Open Group Library.[1] This Guide uses the Cloud Services Model and Cloud Deployment Model definitions from The Open Group White Paper: Cloud Computing Explained (see Referenced Documents) and replicated in Appendix E for convenience.

Re-platforming an application to a cloud platform has many considerations common with virtualization as cloud platforms are generally underpinned by virtualization in order to deliver scalability and elastic provisioning. However, the tools needed for migration and subsequent management of the application will vary depending on whether the service model chosen is PaaS or IaaS, and how a cloud provider enables access to their platform. For example, some cloud providers do not allow Application Platform Interface (API) access to their platform and restrict platform management to a set of tools only provided by them, which may limit the potential to use automation tools.

Where a cloud provider allows access to platform management, the choice of the type of tooling used can impact the skill sets, and therefore the training of IT personnel. For example, a vendor-neutral tool that abstracts the complexity of the underlying infrastructure and technology used will require a different skill set than a tool that operates at the lower level.

There are three important considerations for data when hosting within the cloud: the security of the data at rest, the security of the data in transit, and the geographic location of the data.

The security of the data at rest concerns who can access the data, from where, and includes access to the data outside of the application by cloud provider personnel; e.g., when performing routine operational tasks such as taking backups. Use of encryption for sensitive data is one additional control that can be applied to the data, and requiring that access is through a secure connection mechanism is another control. Other controls include requiring all cloud provider personnel who have access to the data to be suitably vetted, and auditing of data accessed by cloud provider personnel.

The security of the data in transit concerns how the data is secured whilst it is being operationally managed and exchanged; it includes payloads made available via interfaces with other applications, any replicas made, and copies taken as part of backup/restore operations. Typical controls include payload encryption, authentication of endpoints with other applications, and exchange of TLS certificates or mutual TLS certificates.

[1] Refer to www.opengroup.org/library.

The geographic location of the data, and any replicas and backup copies, may be constrained by business or legal requirements that not only impact where the data can be stored but also from where the data can be accessed. This will require assurance from the cloud provider, backed by periodic auditing carried out by the enterprise.

5.7 Refactoring an Application

The decision to refactor an application occurs because it is assessed as having high application health and offering high business value; i.e., it appears in the upper-right quadrant in the 5Rs visualization diagram (Figure 3) but it is recognized that the application will benefit from improving the internal structuring of its code and data to be more efficient.

Refactoring an application does not change its overall architecture, as the focus is on the internal structure of code and data. Externally exposed services and interfaces do not change as part of refactoring, so existing interoperability with other applications is not impacted. The aim of refactoring is to lower maintenance costs by addressing any residual technical debt, to increase maintainability by streamlining internal structures, and to increase adaptability by reorganizing internal components.

Deciding whether an application should be refactored will depend on one or more of the measures listed in Appendix B. For example, measures such as Application Maintenance Costs (AC.MS04), Number of Change Requests Last Year (AC.MS10), Application Architecture (AC.MS16), and Application Adaptability (AC.MS17) will provide metrics that will help determine whether an application will benefit from refactoring.

The first primary consideration for refactoring an application is the change to the code and the internal structure of the code within the application (see checklist item AC.CK02 Application Change in Table 4). Refactoring can be managed in a single large upgrade, or be applied iteratively with small upgrades deployed continuously over a period of time to ensure business activities are not adversely impacted (see checklist item AC.CK07 Business Continuity in Table 4). Iterative upgrades may drive the need to adopt automated deployment tools that better support many frequent deployments of small changes (see checklist item AC.CK05 Technology Change in Table 4).

The second primary consideration for refactoring is the transformation of the data when internal data structures (held in code and data stores) are restructured (see checklist item AC.CK01 Data Change in Table 4). This includes ETL activities to extract, transform, and load data into the revised data structures, and may require the introduction of ETL tools into the technology landscape (see checklist item AC.CK05 Technology Change in Table 4).

Another important aspect of refactoring is to ensure that the external behavior of the application is the same as before (see checklist item AC.CK07 Business Continuity in Table 4) and that it continues to provide the same interoperability with other applications. This is achieved by undertaking full regression testing, and if changes are being made incrementally, may drive the need to adopt automatic testing tools to support such continuous refactoring and improvement (see checklist item AC.CK05 Technology Change in Table 4).

Any introduction of new tools to support iterative development, testing, and deployment would also impact training (see checklist item AC.CK10 Training in Table 4).

5.8 Re-Architecting an Application

The decision to re-architect an application occurs because it is assessed as having low application health and offering high business value; i.e., it appears in the lower-right quadrant in the 5Rs visualization diagram (Figure 3). It should be noted that such applications can also be candidates for modernization work, which goes beyond both the refactoring and re-architecting scope.

Deciding whether an application should be re-architected will depend on one or more of the measures listed in Appendix B. For example, measures such as Number of Change Requests Last Year (AC.MS10), Number of Incidents Last Year (AC.MS20), Number of Unplanned Downtimes Last Year (AC.MS21), Documentation Unavailability Risk (AC.MS12), and Application Complexity (AC.MS15) will provide metrics to decide whether an application needs re-architecting.

Activities typical for re-architecting an application are to:

- Identify any changes to components based on measures related to adaptability, stability, and security
- Include constraints set by principles, sourcing strategy, corporate standards, and compliance requirements to set the game plan for re-architecting
- Create a set of possible solution alternatives based upon Architecture Principles
- Choose one solution alternative
- Assess impact of chosen solution alternative on other existing and planned projects
- Populate and update the Target Architecture to include new components and services
- Remove from the Target Architecture those technology services and components that are being replaced
- Generate or update views describing application cooperation, application portfolio, application structure, application behavior, application information, technology portfolio, and technology usage

Application re-architecture is a wide-ranging subject area and it is impossible to cover every type or kind of re-architecting activity that an application may undergo. Therefore, this section discusses application re-architecting considerations as high-level topics, and references considerations already raised in previous sections.

Improving information system service qualities (such as availability, reliability, and accessibility) can be driven by the need to make an application more adaptable and more accessible to the business. For example, the introduction of a new business operating model, new strategies, new goals, or a major change to the existing model, strategies, or goals can impose different usage and access patterns than in the baseline use of an application.

Uplifting an application to improve service qualities may require using modern technologies and infrastructure. The uplift may be a straightforward exercise, such as ensuring all versions of a technology product are the latest supported version, or be a more complex exercise, such as distributing application components across infrastructure, or moving the components to the cloud.

Another common driver for re-architecting an application can be that the cost of maintenance is disproportionally high for the functionality the application provides. Restructuring the internal components of the application, such as breaking it into smaller self-contained parts, or simplify the underlying Application Architecture, can address this. Use of architecture styles and patterns (for example, event-driven design, model-driven design, Service-Oriented Architecture (SOA), micro-services, etc.) can be employed (see Referenced Documents).

Making use of new and emerging technologies, and leveraging for business advantage, can also be a driver for re-architecting an application. For example, emergent technologies such as the Internet of Things (IoT) and data lakes, etc. can drive the need to re-architect an application.

Also consider the need to effectively develop, deploy, and run the re-architected and modernized application components. There may be a need to apply new thoughts and supporting technologies based on, for example, the IT4IT™ Reference Architecture, Agile, and DevOps. This will drive the need to implement a new or changed way of working, new organizational structure, and new competences that drives the need to develop and train the IT organization (related to checklist items AC.CK05 Technology Change, AC.CK08 Process Change, AC.CK09 Organizational Change, and AC.CK10 Training; see Table 4).

5.9 Migration Planning

As a reminder, within Phase E: Opportunities and Solutions of the TOGAF ADM, an initial version of the Architecture Roadmap is generated based on a consolidated gap analysis for the all the applications within scope of the Architecture Vision. Although this Guide focuses heavily upon applications, it is important to recognize that the consolidated gap analysis will also include gaps in the Business, Data, and Technology domains, and that these gaps will need to be addressed as a whole.

It is imperative to consolidate gaps across a business capability, or related business functions, to ensure the impact to the business is understood and, where possible, operational impacts kept to a minimum. This means reconciling interoperability requirements at both a business and application level and validating any inter-dependencies that go across function boundaries.

An example of interoperability that has to be reconciled, but is often missed, is "Grey IT/Shadow IT" – business-purchased and business-developed applications directly bought or created by the business to augment and extend application functionality. Such applications should be considered as having dependencies on the applications in the portfolio being consolidated.

To formulate an Implementation and Migration Plan, and help identify potential Transition Architectures, applications within each of the 5Rs can be mapped to the 3T time periods as shown in Figure 6:

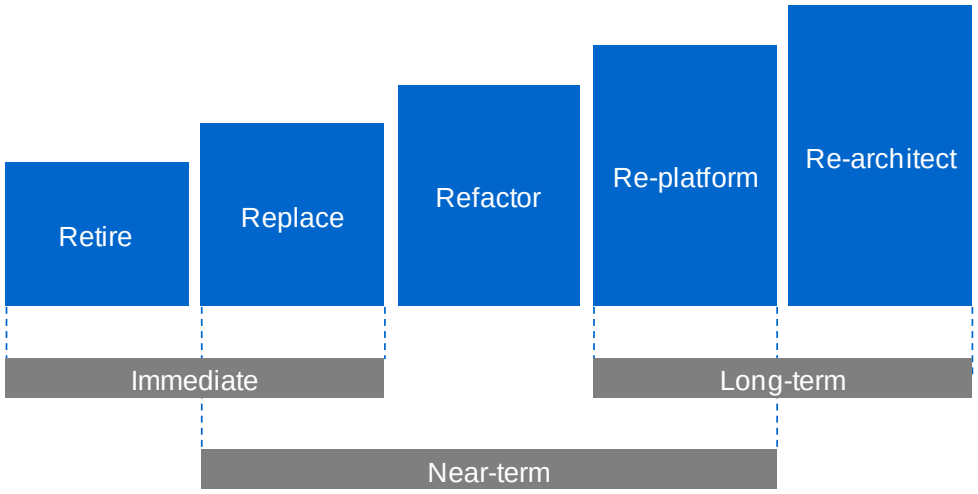

Figure 6: Mapping of 5Rs to Time Periods

As can be seen in Figure 6, there is not a simple one-to-one mapping of the 5Rs to distinct time periods. For example, an application classified as replace may be grouped into either the immediate-term or near-term time periods depending upon a number of factors, including:

- How long it is estimated to migrate data from the application being replaced
- Whether the target application is an existing application or a new application being added to the portfolio
- Required interoperability with other applications

The approach to creating the Implementation and Migration Plan is to handle the implementation of the architecture in three different iterations based on timing of the changes needed. The approach builds on the idea that there is a need for a set of stable states (i.e., Transition Architectures) between the baseline and target state. The base for those stable states is the time needed to implement and deploy suggested changes (i.e., immediate-term, near-term, and long-term).

The more complex the Architecture Roadmap and set of Transition Architectures are, the more critical risk management and a detailed risk assessment become to ensure that the roadmap can be delivered with minimal disruption to the enterprise.

The final placement of elements in the Architecture Roadmap will be refined based on internal priorities, dependencies between elements, visualization of risk, and identification of "quick wins", etc. Figure 7 is an illustrative Architecture Roadmap with simplified content.

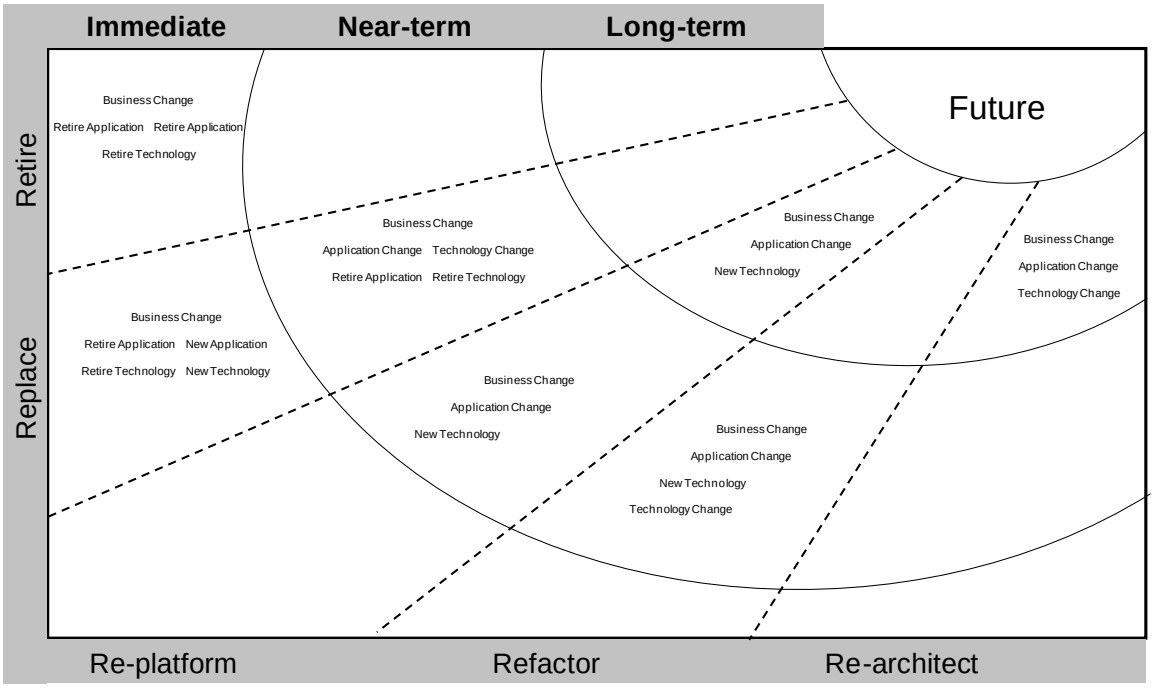

Figure 7: Example Architecture Roadmap

5.10 Summary

The Target Architecture defines the consolidation actions required for each application, the timeframe within which the actions should be performed, and an Implementation and Migration Plan for delivering the architecture.

In identifying the consolidation actions, principles and other grouping criteria will be applied at conceptual, logical, and physical levels to provide alternative Target Application Architectures offering choices in how the consolidation can be achieved, with trade-off analysis informing decision-making.

When creating the Implementation and Migration Plan, use of the 5Rs technique to categorize applications based on their relative health and value to the business, combined with the 3Ts technique to group the applications into distinct time periods, will accelerate the planning exercise.

A Example Stakeholders and Stakeholder Concerns

Table 5 provides a non-exhaustive list of the typical stakeholders who have concerns and interests in application portfolio consolidation.

Table 5: Stakeholders

Reference	Stakeholder
AC.S01	Architecture Board
AC.S02	Program Board
AC.S03	Chief Information Officer (CIO)/Chief Data Information Officer (CDIO)
AC.S04	Corporate Security
AC.S05	Data Guardian/Data Steward
AC.S06	Application Management
AC.S07	Infrastructure Management
AC.S08	Service Management
AC.S09	Application Owner
AC.S10	Business User

There may be other stakeholders who have an interest, but this list represents the usual types of stakeholders who are either accountable, consulted, or informed.

Table 6 provides an example catalog of common concerns that such stakeholders have about application portfolio consolidation.

Table 6: Stakeholder Concerns

Reference	Concern	How Concerns will be Addressed
AC.C01	What applications are in the landscape?	Via views identifying information system services/components
AC.C02	What is the structure of applications?	Via views describing how information system services/components are structured
AC.C03	How do applications interact and collaborate?	Via views describing information system service/component behavior and interaction

Reference	Concern	How Concerns will be Addressed
AC.C04	What information is produced/consumed by applications?	Via views describing how information is used by information system services/components
AC.C05	What business capabilities rely on specific applications?	Via views describing how business activities are supported by information system services/components
AC.C06	What technologies are in the landscape?	Via views identifying technology services/components
AC.C07	What technologies are used by applications?	Via views describing how technology services/components are used by information system services/components
AC.C08	What does it cost to run and maintain an application?	Via views calculating costs for information system services/components
AC.C09	What are the alternative Target Application Architectures?	Via views describing alternatives and criteria
AC.C10	What is the lifecycle for applications?	Via views describing the application lifecycle
AC.C11	How will consolidation be planned and delivered?	Via views describing how the applications in scope will be analyzed and grouped into manageable work packages

B Example Application Health Measures

The following tables provide example measures that can be used to assess the health of an application, or a group of applications. The measures can be collected via stakeholder interviews, by sending out questionnaires to stakeholders, and/or using measures collected from monitoring sources and SLA measures and reports.

Table 7: Business Value Measures

Reference	Measure	Description	5Rs Indicator
AC.MS01	Business Criticality	Valuate the business criticality of the application based on the criticality of the business capabilities that are supported by the application.	Business Value
AC.MS02	Number of Users	Record the number of users using the application component. This is used together with the business criticality measure to valuate the importance of the application.	Business Value
AC.MS03	Regulatory Need	Determine whether there is a regulatory (legal, financial, civil, other) need for the application.	Business Value

Table 8: Cost-Related Measures

Reference	Measure	Description	5Rs Indicator
AC.MS04	Application Maintenance Costs	Collect figures and put together the yearly maintenance budget and the estimated costs over the entire lifecycle. Add figures on the actual outcome for maintenance. This is used to identify applications with high maintenance costs and applications with non-predictable maintenance costs.	Refactor, Replace, Re-architect
AC.MS05	Contractual Cost	Compile the contractual costs associated with the application development and maintenance, including support costs.	Refactor, Replace
AC.MS06	License Cost	Compile the yearly license costs associated with the application.	Re-platform, Refactor, Replace
AC.MS07	Application Operations Budget/Cost	Collect figures on how much it costs to operate the application per year. Calculate budget overruns as a figure on how predictive the operations costs are.	Re-platform, Replace
AC.MS08	Infrastructure Costs	Collect and put together application costs related to the underlying technology infrastructure.	Re-platform, Refactor

Reference	Measure	Description	5Rs Indicator
AC.MS09	FTE Count	Collect the number of FTEs involved in application development and application maintenance.	Retire, Replace, Refactor, Re-architect

Table 9: Maintainability-Related Measures

Reference	Measure	Description	5Rs Indicator
AC.MS10	Number of Change Requests Last Year	Collect statistics on the number of change requests implemented, and their average costs. This helps to identify components that need continual adaptations to fulfill business needs.	Retire, Replace, Refactor, Re-architect
AC.MS11	Source Code Unavailability Risk	Check if the source code is available and versioned.	Replace
AC.MS12	Documen-tation Unavailability Risk	Check what documentation is available and valuate its quality and accuracy.	Retire, Replace
AC.MS13	Product Vendor Support	Check what product support is available from the vendor and valuate the risk of discontinued product support.	Re-platform, Replace
AC.MS14	Skill Paucity	Determine whether it is possible to extend knowledge support and maintenance staff. This helps to predict if there are available resources on the market and the probability of shortage of resources.	Re-platform, Replace

Table 10: Adaptability-Related Measures

Reference	Measure	Description	5Rs Indicator
AC.MS15	Application Complexity	Collect information regarding the number of internal and external integrations and the type of integrations.	Re-architect
AC.MS16	Application Architecture	Collect information about the age of application, application complexity, architecture style, size, ease of change (adaptability).	Re-architect, Refactor
AC.MS17	Application Adaptability	Valuate application adaptability by collecting information about the ease of change; e.g., by using information about time and cost to implement new or changed functionality, by valuating current architecture to identify possible difficulties in adapting to changed business models.	Re-architect, Refactor

Table 11: Lifecycle-Related Measures

Reference	Measure	Description	5Rs Indicator
AC.MS18	Application Lifecycle State	Collect or assign a lifecycle state to the application to identify applications that are near or past the end of their lifecycle.	Retire, Replace
AC.MS19	Technology Platform Lifecycle	Valuate the lifecycle of the underlying technology components; e.g., identify technology that has past or is close to end-of-life.	Re-platform

Table 12: Stability-Related Measures

Reference	Measure	Description	5Rs Indicator
AC.MS20	Number of Incidents Last Year	Collect statistics on the number of incidents per application component, and their severity. This helps to validate how error-prone an application is.	Replace, Re-architect
AC.MS21	Number of Unplanned Downtimes Last Year	Record the number of unplanned downtimes and the average time taken to restore to normal operation. This helps to get an understanding of application complexity and how easy an application is to operate.	Replace, Re-architect
AC.MS22	Service-Level Agreements	Based on SLA measures and reporting, identify application components that don't fulfill agreed service levels.	Replace, Re-architect
AC.MS23	Technology Platform Stability	Collect statistics on incidents, problems, and severity related to the underlying technology platform.	Re-platform

Table 13: Security-Related Measures

Reference	Measure	Description	5Rs Indicator
AC.MS24	Number of Security-related Incidents Last Year	Collect statistics on the number of security incidents per application component and their severity. This helps to validate how error-prone an application is from a security point of view.	Replace, Re-architect, Refactor
AC.MS25	Number of Security-related Change Requests Last Year	Collect statistics on the number of change requests implemented related to security services and their average cost. This helps to identify components that need continual adaptations to fulfill security needs.	Replace, Re-architect, Refactor

Reference	Measure	Description	5Rs Indicator
AC.MS26	Number of Roles Defined	Collect statistics on the number of security roles per application. This helps to validate how complex the access rights settings are to maintain.	Re-architect, Refactor

C Example Model Kinds, Models, Viewpoints, and Views

Please refer to ISO/IEC/IEEE 42010:2011 (see Referenced Documents) for an introduction to model kinds.

C.1 Model Kinds

Modeling supports our understanding of the Baseline Architecture and how it needs to change to meet the Architecture Vision to achieve the Target Architecture.

There are a number of elementary questions that modeling helps to answer:

- What kinds of things do we have, or do we need?
- Where are these things located, or need to be located?
- How do these things interact with other things, or need to interact?
- How are these things structured/composed, or need to be structured/composed?
- On what do these things have dependencies?
- How are these things managed, or need to be managed?
- How are these things secured, or need to be secured?
- How do we transition these things to other things?

Example model kinds, used for illustration, are listed in Table 14.

Table 14: Model Kinds

Name	Description	Format
Location Catalog	Lists the locations within an architecture domain.	List
Business Activity	Describes what business activities are performed by actors in roles.	Matrix, Diagram
Information Catalog	Lists the data entities, logical data components, or physical data components within an architecture domain.	List
Information Security	Describes how information is classified and what controls are required.	List, Matrix, Diagram
Service Catalog	Lists the services within an architecture domain.	List

Name	Description	Format
Service Interaction	Describes how services within an architecture domain interact/communicate with each other.	Matrix, Diagram
Service Composition	Describes the hierarchy of services within an architecture domain.	Matrix, Diagram
Service Realization	Describes which logical and physical components realize services within an architecture domain.	Matrix, Diagram
Service Usage	Describes how services in one architecture domain are used by services in another.	Matrix, Diagram
Service Security	Describes how services are accessed.	Matrix, Diagram
Service Lifecycle	Describes the standards class lifecycle for services within and across architecture domains.	Matrix, Diagram
Component Catalog	Lists the components within an architecture domain.	List
Component Interaction	Describes how components within an architecture domain interact/communicate with each other.	Matrix, Diagram
Component Composition	Describes the hierarchy of components within an architecture domain.	Matrix, Diagram
Component Distribution	Describes where components within an architecture domain are located/distributed.	Matrix, Diagram
Component Usage	Describes how components in one architecture domain are used by components in another.	Matrix, Diagram
Component Security	Describes how components are accessed.	Matrix, Diagram
Component Lifecycle	Describes the standards class lifecycle for components within and across architecture domains.	Matrix, Diagram

C.2 Models

Table 15 illustrates example models and associated model kinds that can be used to address stakeholder concerns.

Table 15: Models

Name/Description	Metamodel Entities	Model Kind
Information System Service Catalog Lists the information system services.	Information System Service	Service Catalog

Name/Description	Metamodel Entities	Model Kind
Information System Service Interaction Describes how information system services interact/communicate with each other.	Information System Service	Service Interaction
Information System Service Composition Describes the hierarchy of information system services.	Information System Service	Service Composition
Information System Service Usage Describes what information system services are used by business services, processes, and capabilities.	Information System Service Business Service Process Business Capability	Service Usage
Information System Service Security Describes how information system services are accessed and by what roles/actors.	Information System Service Role Actor Organizational Unit	Service Security
Information System Service Realization Describes how information system services are realized by application components.	Information System Service Logical Application Component Physical Application Component	Service Realization
Information System Service Lifecycle Describes the standards class lifecycle for information system services.	Information System Service	Service Lifecycle
Application Component Catalog Lists the application components.	Logical Application Component Physical Application Component	Component Catalog
Application Component Interaction Describes how application components interact/communicate with each other.	Logical Application Component Physical Application Component	Component Interaction
Application Component Composition Describes the hierarchy of application components.	Logical Application Component Physical Application Component	Component Composition
Application Component Security Describes how application components are accessed and by what roles/actors.	Logical Application Component Physical Application Component Role Actor Organizational Unit	Component Security
Application Component Distribution Describes where application components are located/distributed.	Physical Application Component Location	Component Distribution
Application Component Lifecycle Describes the standards class lifecycle for application components.	Logical Application Component Physical Application Component	Component Lifecycle

C.3 Viewpoints

Table 16 provides example viewpoints and associated model kinds that can be used when creating the Baseline and Target Architectures for application portfolio consolidation.

Table 16: Viewpoints

Reference	Viewpoint	Model Kind(s)
AC.VP01	**Application Portfolio** Identifies the applications in the landscape.	Service Catalog [c] Component Catalog [c/l/p]
AC.VP02	**Application Structure** Describes the composition of applications.	Service Composition [c], Service Realization [c], Component Composition [c/l/p]
AC.VP03	**Application Behavior** Describes the interactions between applications.	Service Interaction [c], Component Interaction [c/l/p]
AC.VP04	**Application Information** Describes the information produced/consumed by applications.	Information Usage [c/l/p]
AC.VP05	**Application Cooperation** Describes how applications support business activity.	Service Usage [c/l/p], Information Usage [c/l/p], Service Composition [l/p]
AC.VP06	**Technology Portfolio** Identifies the technologies in the landscape.	Service Catalog [c], Component Catalog [c/l/p]
AC.VP07	**Technology Cooperation** Describes how technologies support applications.	Service Usage [c], Component Usage [l/p]
AC.VP08	**Application Costs** Itemizes the costs for running and maintaining applications.	Component Cost [l/p]
AC.VP09	**Alternative Target Application Architectures** Describes any alternative Target Application Architectures identified, the criteria used to determine the alternatives, along with trade-off analysis.	Application Alternative [c/l/p], Trade-off Analysis [c/l/p]
AC.VP10	**Application Lifecycle** Itemizes lifecycle state for applications.	Component Lifecycle [l/p]
AC.VP11	**Consolidation Planning** Describes how applications are classified and grouped for consolidation.	Application Classification [c], Application Grouping [c]

Legend

[c] Model kind is populated and used at the conceptual abstraction level.
[l] Model kind is populated and used at the logical abstraction level.
[p] Model kind is populated and used at the physical abstraction level.

D T-Analysis

A common approach to undertaking an analysis of an existing application landscape is to split the analysis into a number of iterative passes.

This approach can also be used for discovery, where little or no information is known beforehand.

D.1 Initial Pass

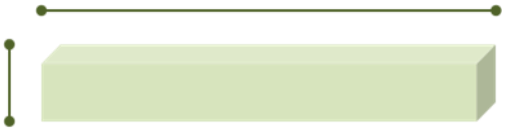

Figure 8: T-Analysis Initial Pass

The initial pass takes a wide breadth across the application landscape but with a shallow depth to determine the scope of analysis or discovery. The information collected from this initial pass will identify the number and types of applications within the landscape.

D.2 Subsequent Passes

All subsequent passes analyzing the application landscape will have a targeted scope with a narrower breadth but a deeper depth. The breadth can be per application, or more frequently a set of related applications grouped based on some criteria – for example, their business value, the business area they service, or the vendors, etc. The depth of the analysis will potentially cover the technology platforms (hardware and software) used to host and support the applications.

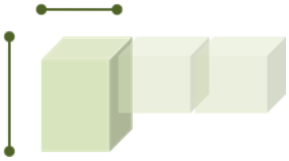

Figure 9: T-Analysis Subsequent Passes

It is important to understand the depth of analysis needed for a particular iteration – too deep a depth can be "modeling for modeling sake" and potentially wasted effort.

The gap analysis technique is a useful tool for managing and controlling the analysis or discovery effort. It can be used to determine what depth of analysis is needed when looking at Baseline and Target Application Architectures within a specific iteration. If the question "what has changed?" between the baseline and target can be answered, then the modeling effort is complete for the current iteration. If the question cannot be answered, then increase the depth of analysis and modeling granularity in the next iteration until the question can be answered.

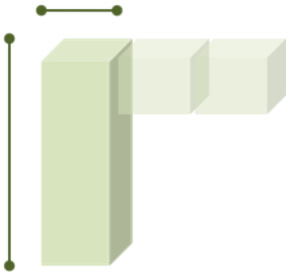

Figure 10: T-Analysis Increasing Depth of Analysis

E Cloud Definitions

The following definitions for Cloud Service Models and Deployment Models are taken from The Open Group White Paper: Cloud Computing Explained and The Open Group Guide: Maximizing the Value of Cloud for Small-Medium Enterprises (see Referenced Documents).

E.1 Service Models

- Software-as-a-Service (SaaS): the consumer does not manage or control the underlying cloud infrastructure, including network, servers, operating systems, storage, or even individual application capabilities
- Platform-as-a-Service (PaaS): the consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications
- Infrastructure-as-a-Service (IaaS): the consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, deployed applications, and possibly limited control of select networking components (e.g., host firewalls)

E.2 Deployment Models

- Private cloud: the cloud infrastructure is operated solely for an organization and may be managed by the organization or a third party, and may exist on-premise or off-premise
- Community cloud: the cloud infrastructure is shared by several organizations and supports a specific community that has shared concerns (e.g., mission, security requirements, policy, and compliance considerations) and may be managed by the organizations or a third party, existing on-premise or off-premise
- Public cloud: the cloud infrastructure is made available to the general public or a large industry group and is owned by an organization selling cloud services
- Hybrid cloud: the cloud infrastructure is a composition of two or more clouds (private, community, or public) that remain unique entities but are bound together by standardized or proprietary technology that enables data and application portability (e.g., cloud bursting for load-balancing between clouds)

Abbreviations & Acronyms

API	Application Platform Interface
CapEx	Capital Expenditure
CDIO	Chief Data Information Officer
CIO	Chief Information Officer
CMDB	Configuration Management Database
COTS	Commercial Off-The-Shelf
CRM	Customer Relationship Management
ETL	Extract, Transform, Load
FTE	Full-Time Equivalent
IaaS	Infrastructure-as-a-Service
IaC	Infrastructure as Code
IoT	Internet of Things
ITSM	IT Service Management
KPI	Key Performance Indicator
NFR	Non-Functional Requirement
OpEx	Operational Expenditure
PaaS	Platform-as-a-Service
PMO	Program Management Office
RPO	Recovery Point Objective
RTO	Recovery Time Objective
SaaS	Software-as-a-Service
SLA	Service-Level Agreement
SME	Subject Matter Expert
SOA	Service-Oriented Architecture
TCO	Total Cost of Ownership
TLS	Transport Layer Security

Index