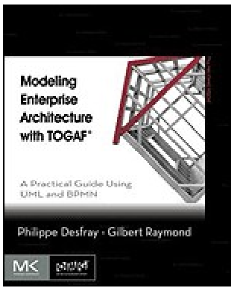


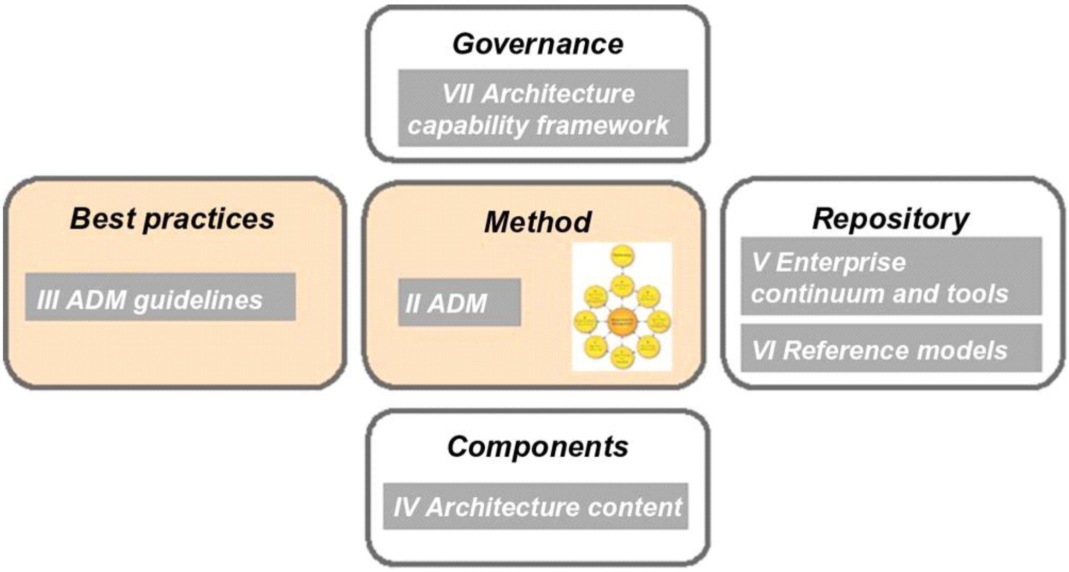
Modeling Enterprise Architecture with TOGAF: A Practical Guide Using UML and BPMN

by Philippe Desfray and Gilbert Raymond

Elsevier Science and Technology Books, Inc.. (c) 2014. Copying Prohibited.

Chapter 2: The ADM Method

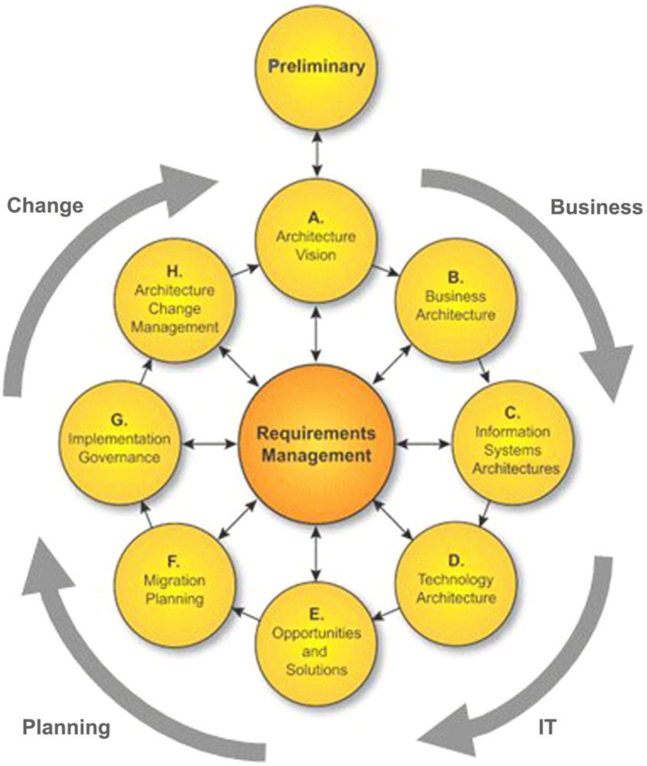
The ADM method constitutes the heart of the TOGAF document, as an enterprise architecture transformation method. This chapter describes how it functions and explains the different phases involved and the relationships between them. It also covers the iterative approach of the method, which should be understood as a guide that can be adapted according to the reality of a given situation. This chapter also looks at best practices associated with the ADM method, which are described in part III of the TOGAF document. The ADM method itself is described in part II.



2.1 THE ADM CYCLE

2.1.1 The Breakdown into Phases

The ADM method defines eight sequential phases (A to H) and two other special phases: the preliminary phase and the requirements management phase. Figure 2.1 shows the most frequently referenced TOGAF diagram, which summarizes this approach through a high-level four-part breakdown: business, information technology (IT), planning, and change.


Preliminary
Change
Business
A. Architecture Vision
H. Architecture Change Management
B. Business Architecture
G. Implementation Governance
Requirements Management
C. Information Systems Architectures
F. Migration Planning
D. Technology Architecture
E. Opportunities and Solutions
Planning
IT

Source: © 2008 The Open Group.

Figure 2.1: Architecture development method (ADM)—TOGAF9

The sequence of phases A to H is broken down as follows:

- Phase A: Vision
- Phase B: Business architecture
- Phase C: Information system architecture
- Phase D: Technology architecture
- Phase E: Opportunities and solutions
- Phase F: Migration planning
- Phase G: Implementation governance
- Phase H: Architecture change management

All phases are described in a similar way:

- The objectives, which define the expected results
- The approach, which provides a guide and recommended strategy
- The input and output, which specify what each phase consumes or modifies
- The different steps, in the form of a breakdown of the work to be carried out

Even though the progression of phases is described in a strictly sequential manner (from A to H), this sequence can be reviewed and adapted according to the context, notably in the form of iterations within the ADM cycle (see Section 2.3). More generally, the "crop circle" diagram should be considered as a reference structure rather than an immutable progression, especially since it remains possible, and even preferable, to question or adjust at any time a part of the results obtained earlier. The identification of new constraints or the reformulation or addition of details to requirements can cause certain new aspects to appear, aspects that were not sufficiently exploited during earlier phases. As an example, the main output document of phase A, "architecture vision," is only definitively validated during phase F. However, high-quality elaboration implies convergent progression, which does not call into question the principles and foundations defined at the outset.

2.1.2 The Typical Path

Figure 2.2 presents an overview of the progression of an ADM cycle, from the preliminary phase right through to phase H. This typical path is guided by one major goal: the need to obtain the expected result by mastering each step of the process. This goal requires rigorous preparation, a description of the target with regard to what already exists for all facets (business, information system, and technology), precise evaluation of the gaps and risks determining the choice of trajectory, and finally evaluation of the results and careful management of any adjustments made.

Note It should be noted that the actual realization of changes falls outside the scope covered by enterprise architecture. Phases G and H are dedicated to implementation governance, notably through the control and follow-up of implementation projects. These implementation projects remain the responsibility of the usual enterprise entities, managed by project managers. The fundamental role of the team in charge of enterprise architecture consists of guaranteeing the conformity of deployed elements with regard to the architecture principles defined for all impacted units.

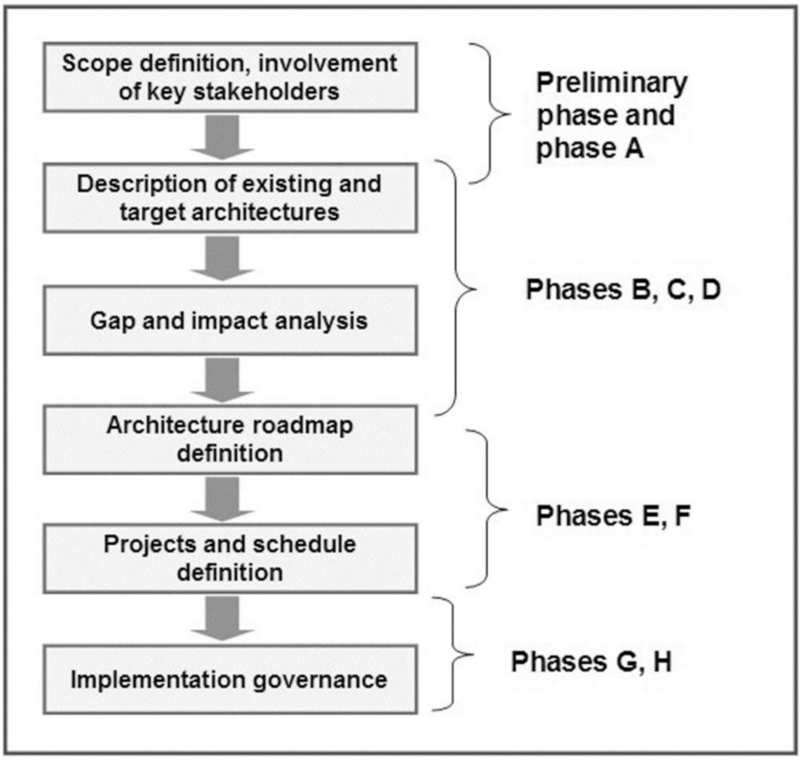

Scope definition, involvement of key stakeholders
Description of existing and target architectures
Gap and impact analysis
Architecture roadmap definition
Projects and schedule definition
Implementation governance
Preliminary phase and phase A
Phases B, C, D
Phases E, F
Phases G, H

Figure 2.2: Typical path of an ADM cycle

TOGAF describes each step of each phase in detail. This does not mean that the realization of each of these steps is systematic. Some results are already available, either because they are produced by other entities or because they are linked to more general activities. For example, if architecture principles are recorded as results from phase A, they can simply be checked where they already exist. The main thing here is that the presence and conformity of each result must be checked, just like a checklist associated with each phase.

2.2 THE PHASES OF THE ADM

2.2.1 The Preliminary Phase

The goal of this phase is to prepare the enterprise for the realization of the architecture work:

- The organization and governance of the architecture
- General principles
- Methods
- Tools
- The architecture repository
- The start of the ADM cycle

These elements directly concern the adaptation of the architecture framework, in other words TOGAF.

In this way, the preliminary phase is not part of an ADM cycle but can be considered at any time during the ADM cycle as part of the evolution of the EA practice, especially in the context of using maturity models as a means of identifying opportunities for transition initiatives. Preliminary phase activities are essentially cross-organizational, linked to the general governance of the enterprise architecture, and their aim is to enable the enterprise to master the management and transformation of its architecture. However, it is during the preliminary phase that the start of a particular ADM cycle is decided upon and prepared for. This is detailed in the "Request for Architecture Work" document, which contains all the elements that form the basis of an enterprise architecture change project (sponsors, strategic goals, constraints, the budget framework, and the strategic plan). This document constitutes a contractual reference guide for the progress of the entire TOGAF cycle itself, from phase A onward.

Finally, TOGAF provides a good summary of this phase in the form of where, what, why, who, and how.[1]

2.2.2 Phase a (Vision)

Phase A is the first phase of the ADM cycle, triggered by the validation of the "Request for Architecture Work" document. Phase A has two main goals:

- First, phase A further develops and enriches elements resulting from the preliminary phase, such as architecture principles, key indicators, and the organization or planning of elaboration work.
- Second, phase A prepares subsequent phases by providing a general representation of the baseline and target architectures. At this stage, these remain high-level representations, whose goal is to highlight structuring points and typical solutions.

Communication plays a key role during this phase. All stakeholders must have the same understanding, in order to obtain consensus on orientations and expected results. Other points are also dealt with, such as the identification of fundamental requirements, their links to strategic goals, or risk management. The "Architecture vision" document constitutes the main output of this phase.

To sum up, at the end of phase A we have a common vision of:

- Organization: the stakeholders, their roles, their respective involvement
- Orientation: a consensus on the principles, goals, major requirements, and constraints
- The scope covered, the most impacted parts
- The roadmap: the ADM cycle development plan, the resources, and the budget allocated
- A macroscopic vision of baseline architecture and target architecture
- Major risks and associated risk reduction actions

In other words, we know where we're going, how we're getting there, and with whom.

Note that at this stage the perspective is horizontal, and covers all architecture domains (business, information system, and technology), unlike the following three phases, which operate vertically, focusing on one particular domain.

2.2.3 Phases B, C, and D (Elaboration of Business, Information System, and Technology Architectures)

Most of the content of the following three phases—B (Business), C (Information System), and D (Technology)—consists in detailing the target and baseline architecture, measuring the gap between the two, and evaluating the impact of change on all facets of the enterprise. The combination of these elements is used to draft the roadmap for transition. This first draft of the roadmap is elaborated progressively throughout phases B, C, and D, and serves as the foundation for phases E and F, which are in charge of defining the transformation plan (Figure 2.3).

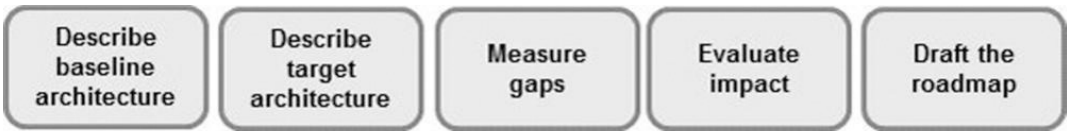


Figure 2.3: Main activities common to phases B, C, and D

Each phase begins with the definition of the views that will be used to materialize the baseline and target architectures. Remember that the goal of these views is to adapt the representations of the architecture to each stakeholder's viewpoint.

Architecture descriptions are consigned to the architecture definition document (central document). This document is enriched during each phase, before being finalized and validated prior to the start of migration work. Concretely, each phase will complete the chapter(s) that concerns it, so that the document spans all architecture domains.

Of course, as well as depending on each situation, the choice of target architecture also integrates recurrent questions. Consequently, TOGAF recommends that the repository be reviewed before each decision in order to reuse the experience accumulated during earlier work wherever possible. This repository review is noted as a "checklist" action at the start of each phase, so as to conform to the norms in place within the enterprise and to promote general harmonization.

Impact assessment should be considered in a cross-organizational way, for two reasons. First, because each phase evaluates its impact beyond its own scope. During phase B, for example, the impact of evolutions on technical elements is also assessed. If, for example, the executive management team decides to remove a product range, it is easy to work out the consequences of this decision on the corresponding database. Second, because the sheer number of relationships within an enterprise can lead to all sorts of unexpected side effects on entities outside the initial scope.

During these three phases, another essential result is expected, namely the definition of requirements. We go into this point in more detail in the chapter dedicated to the requirements management phase (Section 2.2.6). Generally speaking, the aim here is to clearly specify what will be implemented in the target architecture. These requirements are recorded in the "Architecture Requirements Specification" document, which is delivered by each of the three phases. Special attention must be paid to so-called "nonfunctional" requirements, which determine the conditions and limitations surrounding the delivery of services. These limitations have a significant influence on solutions, their feasibility, and their cost, which can draw into question certain choices made earlier.

Phase B (Business Architecture)

The structural similarity of phases B, C, and D should not detract from the determining role of phase B, since it is the business that drives the architecture in all its forms. The formalization of business elements (requirements, processes, entities) is the prelude to all valid logical or technical constructions.

This is all the more true when we consider that the goal of phase B is also to demonstrate the pertinence of the work being carried out. Goals are established during the earlier phases, but it is only when business architecture elements are precisely developed that the target solution can be installed and its consequences observed. For example, the description of modifications carried out on a business process shows the real-life result of these modifications on tasks run by operators, new services to provide, or modifications applied to exchanged information.

In terms of architecture descriptions, phase B mainly concentrates on the following elements:

- Business motivation elements (drivers, goals, objectives)
- Organizational units
- Business functions and services
- Business processes
- Business roles and actors
- Business entities

Business entities describe key business concepts and provide the essential entry point to phase C (in the Data Architecture subphase). Business processes are often the key to understanding an enterprise's real activity, and by extension its architecture.[2]

Phase C (Information Systems Architecture)

Information system architecture is a kind of bridge between the business view and its physical translation. It defines software components (applications and data) that support the automation or realization of business capabilities and functions, without integrating technological realities (this point is discussed in the Phase D part).

Remember that phase C (information system architecture) is itself composed of two subphases: data architecture and application architecture.

These two facets (data and application) are reunited in a single phase because of their proximity in the construction of information system architecture. One of the expected results consists in allocating each data group to one application component, which will handle its management, becoming, as it were, the owner of the data group in question.

Phase D (Technology Architecture)

Unsurprisingly, the role of phase D is to establish the technological and physical correspondence of the elements developed during the previous phases. In particular, technology architecture defines the platforms and execution environments on which the applications run and the data sources are hosted for use.

So what are the links between application architecture and technology architecture? A first approach consists in considering them as two separate elements, so as to avoid any technical "intrusion" into the work of the application architect. The opposite approach would lead us to consider application architecture as a simple reformulation of the technical reality.

A position that is too dogmatic will lead to a dead end: What is the point of developing a "virtual" application architecture with no link to the reality of the deployed applications? Common sense (and purse strings) calls for more realism. Even though it must remain logical, application architecture (including its service-oriented architecture (SOA) formulation) is not completely separate from its physical translation. The most important thing here is the identification of the role of each application or component, independent of its technical implementation: the fundamental structure is similar and the viewpoint is different, just like a logical service interface, which is not fundamentally modified by its implementation in Java or via a web service.

Bearing in mind these two perspectives, a question comes to mind: Should we start by describing the technical architecture or the application architecture? This point is linked to the iterations of the ADM cycle, which will be more generally dealt with in Section 2.3. Remember that the ADM cycle is a generic framework, which does not forbid intrusions into earlier or later phases (the TOGAF document is strewn with suggestions of this type). In practice, no preestablished choices exist: this is the famous choice between "top down" and "bottom up," which always finishes with a compromise. The deployment of external tools imposes a type of architecture that can sometimes have a significant impact on application architecture solutions. In other contexts, architecture will be more oriented by architectural principles, for example to obtain a more progressive structure.

However, let's get back to the result of phase D: the technological architecture, in other words, a coherent set of software components, infrastructures, and technical platforms. These elements can come from external providers or be produced directly by teams within the enterprise. Moreover, the choice between deploying tools that are available in the marketplace or tools resulting from specific developments is a recurrent theme for an enterprise architect. Here too, the repository (see Section 4.1) will assist in this type of choice by making available a set of common norms, patterns, tools, and practices, which will help harmonize solutions within the enterprise.

2.2.4 Phases E and F (Opportunities and Solutions, Migration Planning)

At this point of the ADM cycle, the operational realization of architecture transformation truly begins: projects are set up, schedules defined, resources identified, and operational monitoring put in place. The previous phases have provided the target, an overall roadmap, and now their concrete implementation has to be defined.

Phases E and F look at the scheduling and organization of the implementation of new architecture. Emphasis is placed on building the migration path, which must bring true business benefit to each step.

During phase E, the results of the elaboration phases (B, C, and D) are consolidated: architectures, requirements, and gaps. This consolidation constitutes the raw material used to define transition architectures, while bearing in mind the enterprise's capability for change (for example, new applications to develop and evolutions of existing applications, according to the coverage of the business functions). Technical and organization feasibility, compromises between requirements and costs, and integration constraints are also studied.

Phase F precisely establishes migration scheduling, as well as the constitution of implementation projects and their organization, goals, and costs.

2.2.5 Phases G and H (Implementation Governance, Architecture Change Management)

Phase G establishes the definitive version of architecture contracts with implementation projects, including recommendations from the architecture board. These signed contracts constitute the basis for conformity reviews of implementation projects.

Phase H handles the management of the deployed architecture: change management, including the evaluation of change requests that impact the architecture. It should be noted that certain evolution requests can lead to new ADM cycles.

2.2.6 Requirements Management

What is a Requirement?

TOGAF provides the following definition: "A quantitative statement of a business need that must be met by a particular architecture or work package."

In concrete terms, a set of requirements determines what must actually be implemented, and conversely, what is not retained. Based on given business goals, concrete requirements generally translate how different factors, be they technical, budgetary or organizational, are to be taken into account.

It must be emphasized here that TOGAF advocates a dynamic view of requirements, which are not frozen at the beginning of a cycle but rather can evolve over the course of the project. This is a very important point, since experience has shown that there is often a difference between the initial requirements defined by the business actors and the actual reality of implementation within information systems. Consequently, it is through constant comparison that a solution is developed, in order to take into account all kinds of constraints as early as possible.

Functional Requirements and Nonfunctional Requirements

Based on goals, which are defined in general terms, requirements are usually described in the form of short, precise statements. For example, if the goal is "to provide clients with a mode for ordering online, to replace the current telephone ordering mode," then requirements of the following type will be found: "The client must be able to order a product online at all times."

In actual fact, this requirement contains two requirements of different types:

- A functional requirement: "The client must be able to order a product online."
- A nonfunctional requirement: "It must be possible to place an order at all times."

This distinction between the functional and the nonfunctional is widely recognized today. The functional handles the "what," while the nonfunctional deals with the conditions under which the service is provided. These conditions concern performance, security, availability, reliability, and so on and are the object of detailed listings.[3]

The impact of nonfunctional requirements on architecture is particularly important. For example, for one functional requirement, a high-level reliability requirement will result in the implementation of appropriate means (duplication, dedicated infrastructure, etc.), which will have a significant influence on the architecture of the future system. As a consequence, the way in which nonfunctional requirements are expressed must be particularly well thought out, and above all quantitatively specified wherever possible. In the previous example, the initial formulation proves to be both too vague and too radical. In this case, it would be better to formulate the requirement as follows: "The product ordering system will be unavailable for a maximum of one hour per month" (obviously as far as this corresponds to the actual reality).

Who writes requirements? The most accurate answer to this question is "pretty much everyone." Even if it seems at first glance that responsibility for this task lies primarily with the business side, the previous example illustrates that this activity requires specific skills. While it is true that business needs must not be driven by technical considerations, it is the role of the architect to clarify the formulation of requirements, based on his knowledge of the underlying consequences.

A typical approach is to request an "expert" reformulation of requirements. In this way, we can ensure that requirements make sense to "experts" (in other words, to people specialized in domains above the current domain), that they are taken into account, and that they are feasible according to a reformulation that is accepted by several parties.

The following is an example of a functional requirement formulated during the preliminary phase: "Every client has an account which can be accessed by the account manager."

A business analyst then reformulates this requirement as follows: "An account manager has access to all his clients' accounts. He does not have access to accounts which do not belong to his clients. When a client has no designated account manager, 'cross-organizational' account managers have access to the account, and temporarily play the role of account manager. Conflicts (absence of an account, account manager rights, and so on) are managed by the system administrator in co-ordination with the sales manager."

Here, the business analyst has reformulated the requirement, notably by taking into account concepts and information that were not defined during the preliminary phase. The concepts of "administrator" and "cross-organizational account manager" and information on rights were only defined during phase B, thereby allowing the requirement to be reformulated more precisely.

Centralized Requirements Management

Appearing as it does in the center of the crop circle diagram, requirements management occupies a special place in the ADM cycle. It applies to all phases of the ADM, yet is considered to be independent of each. This choice is explained by the fact that requirements management, as we have just seen, requires particular know-how, independent of the domain in question. Moreover, requirements are not defined according to the type of architecture, since they express a view that is external to the system, and constitute an inseparable whole. Thus, all requirements are analyzed together, using dedicated tools and techniques.

Requirements management is an activity involving the rationalization, hierarchical organization, and monitoring of a set of requirements grouped together within a dedicated repository. This repository is not frozen, and can evolve during the different phases, each of which can add, further define, or invalidate certain requirements.

Figure 2.4 illustrates the overall functioning. Each phase of the ADM produces or modifies requirements, which are then collected, qualified, and organized into a hierarchy by requirements management. These requirements then serve as input for other ADM phases, during which they are analyzed and their impact on architecture determined. This permanent roundtrip encourages an objective view of requirements (for example, removal of repetition), as well as facilitating homogeneous formulation and maintaining overall consistency.

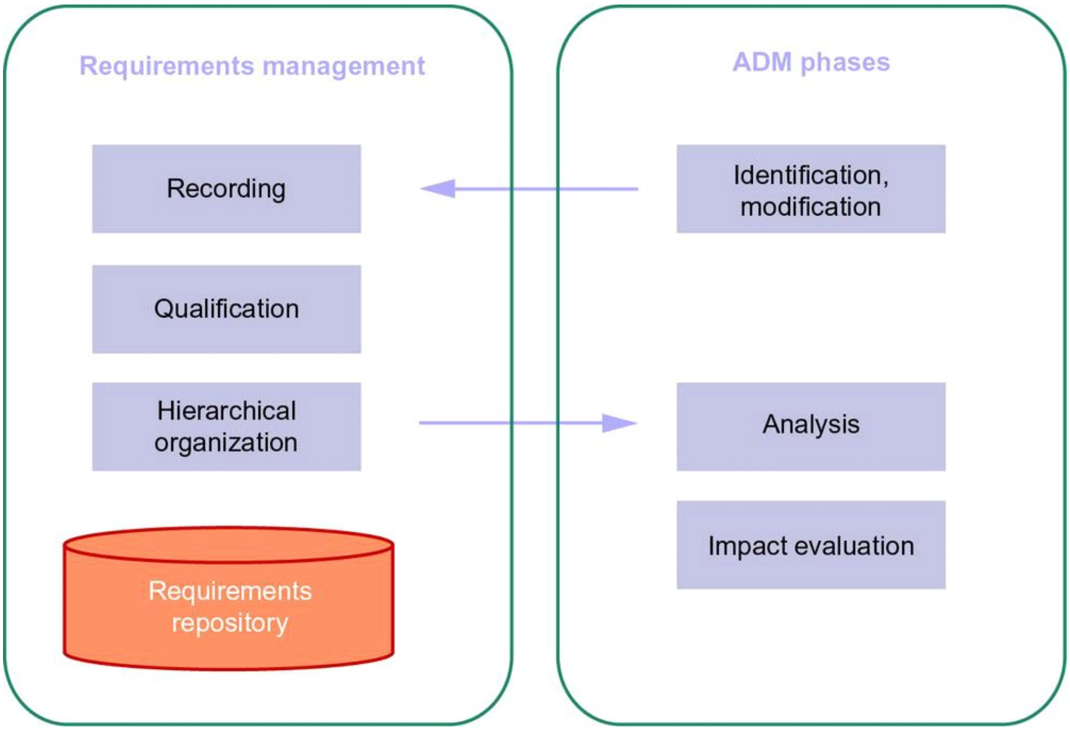


Figure 2.4: Dialog between requirements management and ADM phases

Phase A provides an initial list of requirements, most of which are not described in detail. During phases B, C, and D, certain requirements will be specified, while others will be added according to the type of architecture in question (business, system, or technical). Business requirements are obviously central. Developed during phase B, these requirements are implemented from a system and technical standpoint during phases C and D.

Phase E reviews and consolidates the set of requirements, with particular focus on requirements linked to interoperability. Functional requirements are allocated to the different transition states.

Phase G establishes requirements related to scheduling and divides responsibility for requirements between the different implementation projects.

Requirements are consigned to the "Architecture Requirements Specification" deliverable, which provides a view of the state of the requirements repository at a given point of the ADM cycle.

Requirements Management Techniques

Requirements management is a domain in its own right, and has been the subject of several published works resulting from the fields of software development or business analysis (Volere,[4] BABOK[5]) and systems engineering (SysML[6]).

The scope of enterprise architecture is appreciably different. However, tried and tested techniques can be used and adapted. An enterprise architect is not responsible for writing application specifications documents. However, the formulation of the main requirements, which concretely translate goals, does orient architecture choices.

Probably the most useful technique is that of the "requirement list," which consists of breaking down requirements into basic statements, each of which has a set of properties. These properties help organize requirements according to numerous criteria, and make it easier to objectively analyze them. We have already used a statement of this type with the example "The client must be able to order the product online."

Each basic requirement facilitates the identification and maintenance of links to other architecture elements (goals, processes, application components, etc.). This structured set in a tooled repository constitutes a precious decision-making aid. Several examples of this type are available in Section 7.4.

Business Scenarios

Business scenarios derive the characteristics of architecture directly from the high-level requirements of the business. They are used to help identify and understand business needs, and thereby to derive the business requirements that architecture development has to address.

A business scenario describes:

- A business process, application, or set of applications that can be enabled by the architecture
- The business and technology environment
- The people and computing components (called "actors") who execute the scenario
- The desired outcome of correct execution

A good business scenario is representative of a significant business need or problem and enables vendors to understand the value that a developed solution brings to the customer organization.

A business scenario is essentially a complete description of a business problem, both in business and in architectural terms, which enables individual requirements to be viewed in relation to one another in the context of the overall problem. Business scenarios also play an important role in gaining the buy-in of these key personnel members to the overall project and its end-product: the enterprise architecture.

Business scenarios exist to elucidate requirements, show feasibility, and show how the new architecture will enable goals and requirements to be properly supported.

Business scenarios can be prototypes, but can also be descriptions that show, for example, that a new component combined with the existing application will facilitate the essential parts of a business process.

[1] TOGAF9, chapter 6.2: This preliminary phase is about defining "where, what, why, who, and how we do architecture" in the enterprise concerned.

[2] We look at these two points in detail in Chapter 12.

[3] See ISO 9126.

[4] www.volere.co.uk/template.htm.

[5] www.iiba.org/imis15/IIBA/Home/IIBA_Website/home.aspx.

[6] www.sysml.org.

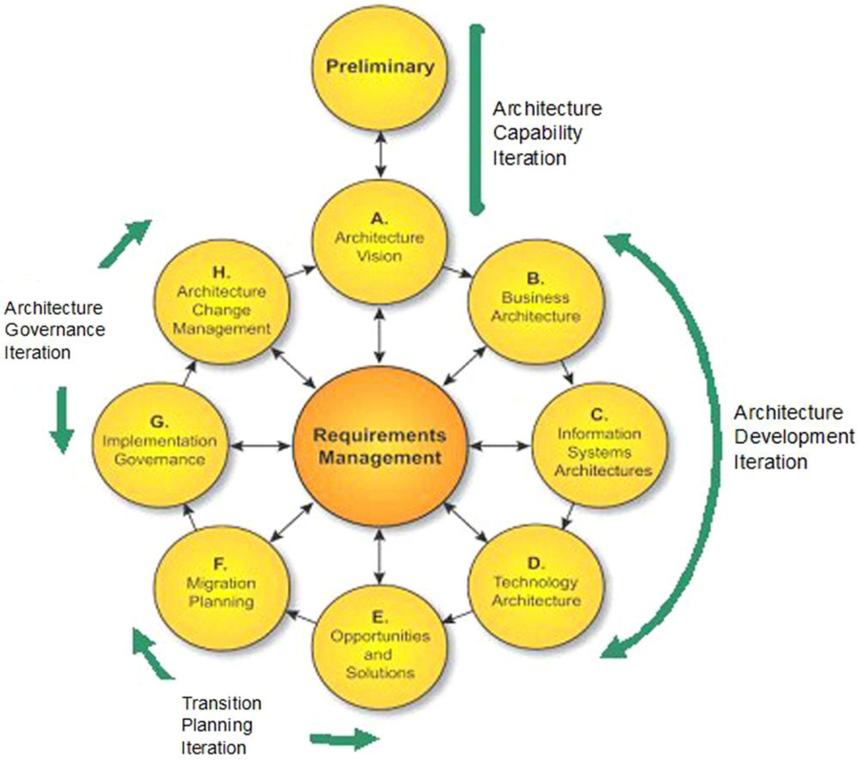
2.3 ITERATIONS

2.3.1 Iteration Cycles

TOGAF strongly recommends the iterative approach and provides a set of best practices and advice on this subject.[7] For example, TOGAF proposes four iteration cycles, based on a grouping of phases:

- Architecture capability iteration, which groups the preliminary phase and the vision phase (phase A).
- Architecture development iteration in the business, system and technological fields, during phases B, C, and D, respectively.

- Transition planning iteration, made up of phases E and F.
- Architecture governance iteration, dedicated to implementation and monitoring during phases G and H (Figure 2.5).


Preliminary
Architecture Capability Iteration
A. Architecture Vision
B. Business Architecture
C. Information Systems Architectures
D. Technology Architecture
E. Opportunities and Solutions
F. Migration Planning
G. Implementation Governance
H. Architecture Change Management
Requirements Management
Architecture Governance Iteration
Architecture Development Iteration
Transition Planning Iteration

Source: © 2008 The Open Group.

Figure 2.5: ADM and iterations—TOGAF9

Typically, a cycle can run several development iterations (phases B, C, and D) in order to successively deal with business architecture, information system architecture, and technological architecture, before starting the transition and planning phases (E and F). This can result in the following phasing:

- Vision phase
- Iteration 1 (Business1, System1, Technology1)
- Iteration 2 (Business2, System2, Technology2)
- Iteration 3 (Business3, System3, Technology3)

2.3.2 Priority to Target Architecture or Baseline Architecture

These choices are partially guided by the perceived value or relevance of the existing architecture: priority can be given to the baseline architecture, or conversely to the target architecture.

For example, the first development iteration cycle (B1, C1, D1) will be dedicated to describing the existing architecture on all levels (business, information system, and technological), while the solution (the target) will only be outlined. The second iteration (B2, C2, D2) will focus particularly on the development of the target architecture on each of these three levels.

The opposite choice is also possible, in other words, focusing first on the target architecture in a BCD cycle and then concentrating on the existing architecture during a later iteration. This technique can be useful when we want to work quickly on solutions, or if a large-scale revision of the existing architecture has been planned.

[7] TOGAF9, chapter 19.

2.4 ADM TECHNIQUES AND GUIDELINES

Part III of TOGAF (*ADM Guidelines and Techniques*) is presented as a kind of "Swiss army knife," whose different parts are used according to our different needs. Most known themes related to enterprise architecture are found here, with recommended references on the subjects dealt with.

2.4.1 The Different Techniques

These different techniques (there are 14 of them) can be categorized as follows:

- Techniques linked to the organization and management of participants:
    - Stakeholder management
    - Business transformation readiness assessment
- Information system architecture techniques:
    - Architecture patterns
    - Architecture principles
    - Using TOGAF to define and govern SOAs
    - Interoperability requirements
    - Security architecture
- Techniques linked to architecture development:
    - Business scenarios
    - Gap analysis
- Techniques linked to the planning and deployment of the target architecture:
    - Migration planning techniques
    - Capability-based planning
- TOGAF adaptation techniques:
    - Applying iteration to the ADM
    - Applying the ADM at different enterprise levels
- Cross-organizational techniques:
    - Risk management

Given the density of certain themes would warrant a whole book in themselves, TOGAF provides a widely accepted summary of techniques, along with references and standards on the subject. This is notably the case with risk management and SOA. More generally, the aim is to provide a kind of "method of the method," enabling each theme to be appropriated in order that we can build our own guidelines from the examples provided.

2.4.2 Techniques in ADM Phases

In the chapters dedicated to the different techniques, TOGAF links each technique to the ADM phases during which it is the most useful. Table 2.1 shows which techniques are used during which ADM phases. Techniques are also identified.

Table 2.1: Use of Techniques in ADM Phases

Techniques	Phases[a]	
Stakeholder management	Preliminary phase Phases A, E, and F	M
Business transformation readiness assessment	Preliminary phase Phases A, E, and F	R
Architecture patterns	Phases A, B, C, and D	S
Architecture principles	Preliminary phase Phase A	R
SOA	Phases B, C, and D	S
Interoperability requirements	Phases A, B, C, D, E, and F	R
Security architecture	All phases	R

Business scenarios	Phases A and B	R
Gap analysis	Phases B, C, and D	R
Migration planning techniques	Phases E and F	R
Capability-based planning	Phases E and F	R
Applying iteration to the ADM	Preliminary phase Phase A	M
Applying the ADM at different enterprise levels	Preliminary phase Phase A	R
Risk management	All phases	R

[a] We have indicated here the main phases during which each technique is used.

M, mandatory; R, recommended; S, supported.

We have chosen not to explain each technique in detail. For more information, readers can refer to the TOGAF document (part III). However, certain techniques are discussed in certain chapters of this book:

- Section 1.2.2: Gap analysis, capability-based planning
- Section 1.2.5: Stakeholder management, business transformation readiness assessment
- Section 12.1: Service-oriented architecture
- Section 2.3: Applying iterations to the ADM
- Section 2.2.6: The technique of business scenarios
- Section 12.3: Interoperability requirements

2.5 FUNDAMENTAL CONCEPTS

The following fundamental concepts were introduced in this chapter:

- ADM cycle: Step-by-step approach and method to transform an enterprise architecture.
- ADM phase: Main stage of the ADM cycle, described by its goals, content, inputs, and outputs.
- ADM iterations: ADM cycle path with repetition of certain phases. TOGAF recommends the use of iterative paths in order to encourage the flexibility and adaptation of the ADM cycle.
- Business scenario: Prototype or model of a subset of the system, made up of a business process and a set of software components or applications, and of all the technical and organizational elements necessary to attain the desired result. Used to validate options and to verify the feasibility of a solution.