ArchiMate® Extension for Modeling and Managing Motivation, Principles and Requirements in TOGAF™

A White Paper by:

Dick Quartel, Wilco Engelsman, Henk Jonkers

October 2010

Table of Contents

Summary

Requirements Management is an ongoing process that plays a central role within the TOGAF ADM. The Requirements Management process is related to all the other ADM phases.

However, the guidance that TOGAF offers for actually doing requirements management is limited. This white paper discusses a methodology and modeling support for requirements management that is well-aligned with both TOGAF and the ArchiMate enterprise architecture modeling language.

In TOGAF, architecture principles govern the architecture process or the implementation of the architecture. They are primarily established in the Preliminary Phase and in Phase A (Architecture Vision) of the ADM. Principles and requirements are strongly related:

- Principles represent general constraints that apply to all designs in a certain context.
- When applied to the design of a specific solution, the applicable principles are specialized into concrete requirements.

In this white paper, we describe:

- A *way of thinking*, which describes the philosophy, principles and assumptions underlying the way requirements management is approached.
- A *way of working*, which describes a reference method that identifies and structures requirements management activities in relation to the TOGAF ADM phases, and facilitates the re-use of existing requirements engineering techniques to support these activities.
- A *way of modeling*, which describes concepts for modeling goals, architecture principles, requirements, stakeholders, their concerns and assessments of these concerns, as well as the relationships between these concepts.
- A *way of supporting*, which describes techniques for viewing and analyzing requirements models and tool support for creating and working with these models.

Introduction

Requirements management is an important activity in the process of designing and managing enterprise architectures. The goals of various stakeholders form the basis for any change to an organization. These goals need to be translated into requirements on the organization's architecture, which should reflect how the requirements are realized by services, processes and software applications in the day-to-day operations. Therefore, the quality of the architecture is largely determined by the ability to capture and analyze the relevant goals and requirements, the extent to which they can be realized by the architecture and the ease with which goal and requirements can be changed.

Nonetheless, the guidance that enterprise architecture methods offer to support requirements management is limited. Furthermore, many modeling techniques focus on *what* the enterprise should do by representing 'as is' and 'to be' architectures in terms of informational, behavioral and structural models at the different architectural layers (e.g., business, application and technology). Little or no attention is paid to represent (explicitly) the motivations and intentions, i.e., the 'why', behind the architectures in terms of goals and requirements.

Principles and requirements are strongly related. Architecture principles can be considered as general requirements that govern and constrain *all* possible designs of some enterprise architecture in a given context. In contrast, requirements constrain and shape *one specific* design of some enterprise architecture. This corresponds to the two commonly used views on enterprise architecture: (i) as the *structure* of some organization in terms of its components and their relationships, and (ii) as a *set of principles* that should be applied to any such a structure.

The first view on enterprise architecture concerns a single design of the organization, whereas the second view concerns any possible design. Requirements are associated with the first interpretation, whereas principles are independent of a specific design and have to be specialized into requirements in the process of designing the organization's architecture. This makes the application of principles an important part of requirements management.

Inadequate requirements management is one of the main causes of impaired or failed IT projects [7], due to exceeding budgets or deadlines, or not delivering the expected results. As mentioned by Brooks in the context of software architecture [2]: "No other part of the work so cripples the resulting system if done wrong". Therefore, the requirements management process and the architecture development process need to be well-aligned, and traceability should be maintained between requirements and the architectural elements that realize these requirements. In this white paper we focus on requirements management and principles for enterprise architectures.

In TOGAF's Architecture Development Method (ADM) [10], requirements management is a central process that applies to all phases of the ADM cycle. However, TOGAF merely presents 'requirements' on requirements management, and refrains from mandating or recommending existing methods and tools from the area of requirements engineering. Different methods and tools may fit different situations, and a choice among them should therefore be left open.

Nonetheless, we argue that the requirements management process in TOGAF should provide more structure to facilitate its application in the ADM. Therefore, this paper presents a method that is based on a generic ‘requirements engineering cycle’. This cycle can be repeated in each phase of the ADM and consists of three steps: (i) problem investigation, (ii) proposition of (alternative) solution(s), and (iii) selection and validation of a solution.

In addition, we argue that the ArchiMate language [6] should be able to support the modeling of goals, requirements and principles. ArchiMate complements TOGAF by defining a modeling language [8]. However, version 1.0 of ArchiMate does not provide concepts for requirements management. Therefore, this paper proposes such concepts, including their relationships to the ArchiMate core concepts.

Finally, a method and language for requirements management should be accompanied by viewpoints and analysis techniques in order to facilitate the communication, documentation and reasoning about motivation models[1] and the requirements management process. Therefore, this paper also discusses some basic motivation modeling viewpoints and associated analysis techniques.

Problem chains – Way of thinking

Requirements engineering (RE) is concerned with the process of finding a solution for some problem. This concern can be approached from a *problem-oriented view*, which focuses on understanding the problem, and a *solution-oriented view*, which focuses on the design and selection of solution alternatives.

The problem-oriented view on RE stems from systems engineering and is about investigating a problem domain. Within this view the requirements engineer describes and analyzes the experienced problems and the relationships between these problems. A popular technique within this view is Goal-Oriented RE (GORE). Goals are identified to address the identified problems, initially defined at a high level, and subsequently decomposed into sub-goals that are more concrete and measurable. Such concrete goals make precise what stakeholders want or need to solve their problems, while abstracting from how this can be achieved.

The solution-oriented view on RE stems from software engineering and is about specifying a solution for some problem. Within this view the requirements engineer specifies the context in which some system will operate, provides a list of desired system functions, and provides a list of quality attributes of those functions. This is the more traditional view on RE. In the context of enterprise architecture, the solution-oriented view corresponds to the specification of requirements on the components of the organization, such as business services, processes and software applications. These requirements should be derived from the goals that are set by the organization, i.e., the responsible stakeholders.

[1] In the sequel, the term ’motivation model’ is used to denote models of goals, requirements and/or principles.

Therefore, the problem-oriented and solution-oriented view could be considered as subsequent phases in a design process.

When applying these views iteratively we can identify so-called *problem chains*. Each chain links a problem to a solution such that the solution is considered again as a problem by the next chain. For example, a business analyst may investigate a business problem and specify a business solution for this problem. This new solution most likely needs IT support, therefore becoming a problem for the IT analyst. Figure 1 illustrates the notion of problem chains.

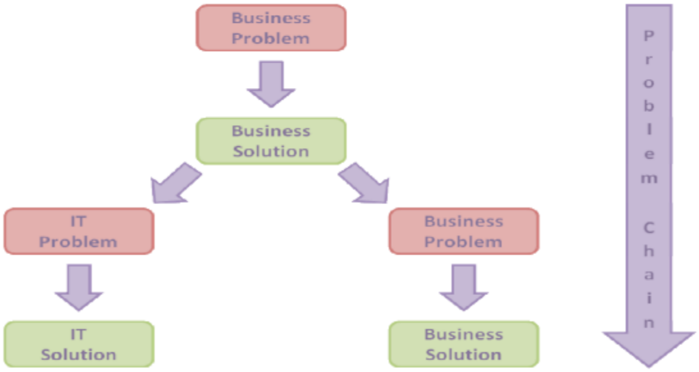


Figure 1. Problem chains

Problem chains link requirements engineering to enterprise architecture. This is illustrated in Figure 2. The *why* column represents the problems, which are addressed through the definition of business goals and requirements. The *what* column represents the solutions, which are defined in terms of enterprise architecture elements such as services, processes and applications. These architecture elements define *what* the enterprise must do to realize the business goals and requirements. At the same time, these requirements engineering elements motivate *why* the enterprise architecture is defined the way it is.

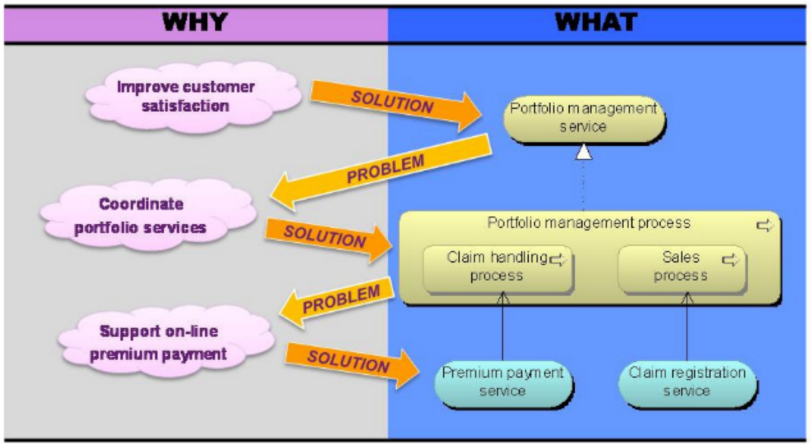

WHY
WHAT
Improve customer satisfaction
SOLUTION
Portfolio management service
PROBLEM
Coordinate portfolio services
SOLUTION
Portfolio management process
Claim handling process
Sales process
PROBLEM
Support on-line premium payment
SOLUTION
Premium payment service
Claim registration service

Figure 2. Linking requirements and architecture

For example, an organization may experience dropping sales, unsatisfied customers and a high workload for the customer support department. In order to address these problems, the organization decides to introduce a new business service to support on-line portfolio management. This solution leads to a new design problem: the creation of new or the adaptation of existing business processes that support this service. Similarly, these processes may require IT support, which leads to the development of new application services.

In TOGAF, architecture principles govern the architecture process or the implementation of the architecture. They guide and constrain the design and implementation decisions taken during the development process. This is done by describing properties that are desired or required from architecture and implementation solutions. In this respect, principles resemble requirements.

However, an important distinction between requirements and principles is that the former are defined during the development process to model the desired properties of the solutions that are conceived, whereas the latter are assumed to be defined beforehand to model desired properties of solutions in general, given a certain context. This means that principles are more general and need to be specialized into requirements to enforce that solutions conform to principles. Principles are intended to be more stable than requirements in the sense that they change less frequently. Organization values, best practices and design knowledge may be reflected and made applicable in terms of principles. For example, frequently occurring (patterns of) requirements may be generalized into principles. Consequently, whereas requirements may be derived as specializations of principles, principles may be derived as generalizations of requirements.

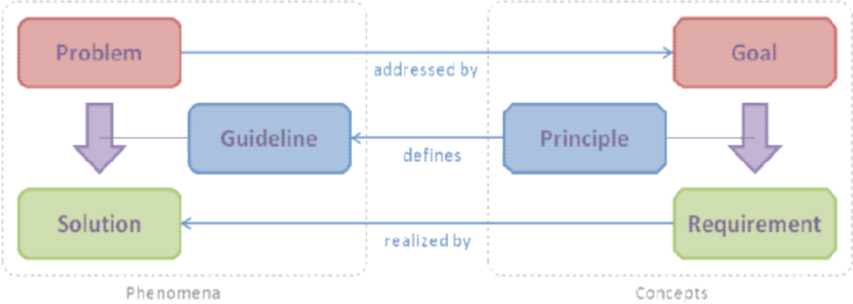


Figure 3. Principles as guidelines

Figure 3 illustrates the relation between goals, principles and requirements. Goals are defined to address some problem. These goals are translated into requirements on a possible solution. This translation from goal to requirements – and thus from problem to solution – is guided by principles.

Figure 4 illustrates the effect of a principle on the design process. Given the goal to improve customer satisfaction, one may suggest requirements for distinct solutions that realize this goal. Together these solutions span up the solution space for this goal and its underlying problem. A principle constrains the solution space by defining desired properties these solutions should have. In this example, solutions should be customer facing. This eliminates solutions that do not support this property.

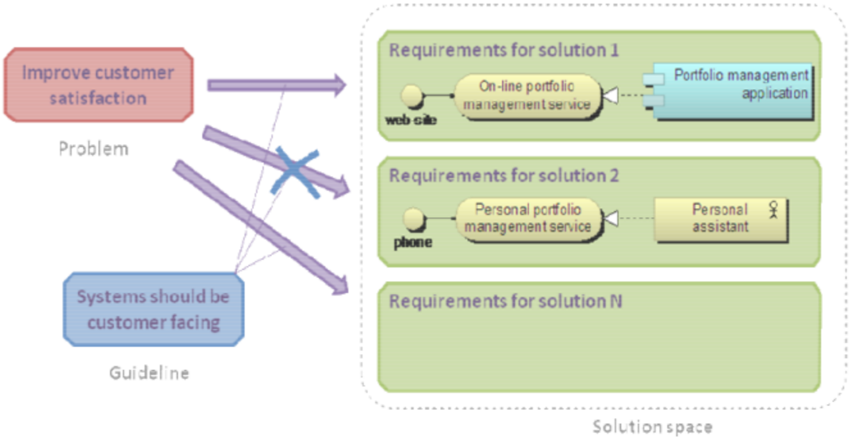

Improve customer satisfaction
Problem
Systems should be customer facing
Guideline
Requirements for solution 1
web site
On-line portfolio management service
Portfolio management application
Requirements for solution 2
phone
Personal portfolio management service
Personal assistant
Requirements for solution N
Solution space

Figure 4. The effect of principles on the design process

This way of thinking enables traceability. Enterprise architecture elements can be traced back to the goals, principles and requirements that motivated their introduction. Reversely, requirements-related elements can be traced forward to the services, processes and applications that implement these elements. This traceability is needed to successfully analyze and manage the impact of changes to an enterprise. For example, in case certain business goals are affected by changes in legislation it becomes now possible to determine precisely which products and services are affected by these changes.

Motivation and architecture – Way of working

The idea of problem chains advocates an iterative way of working. In the context of enterprise architecture, each chain (iteration) translates the current architecture model into a more detailed one that solves some (design) problem. This starts with an investigation of the problem and the elicitation of goals that address the problem. Subsequently, possible solutions are identified by refining these goals, guided by principles, into requirements on existing or new elements of the enterprise architecture, such as the products, services, processes and applications of the organization. In case of alternative solutions, a trade-off analysis may be needed to determine the one that best satisfies the goals that have been set.

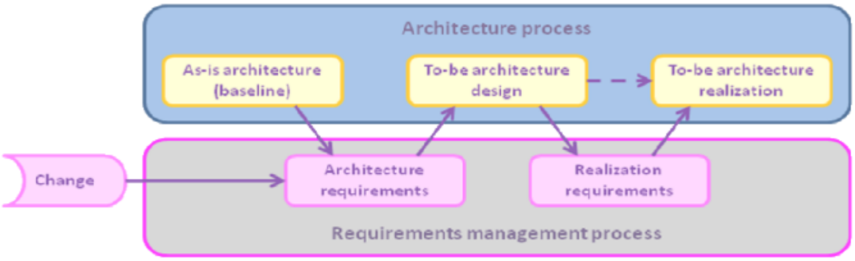

Architecture process
As-is architecture (baseline)
To-be architecture design
To-be architecture realization
Change
Architecture requirements
Realization requirements
Requirements management process

Figure 5. Relation between enterprise architecture and requirements management

Figure 5 illustrates the relationship between motivation models and architecture models, and indirectly, the relationship between the requirements management and architecture processes. *Requirements management* encompasses requirements engineering, i.e., identifying, analyzing, refining and modeling goals, principles and requirements, and the traceability among these elements. Enterprise architecture represents the process of translating requirements into architecture models. These processes are typically divided into distinct phases, which results in a series of motivation and architecture models such that models in succeeding phases refine models from preceding phases (as represented by the dashed arrows). For example, figure 5 illustrates a process of two phases: the design and realization of some enterprise architecture. These phases can be divided again into sub-phases (or steps).

Requirements engineering cycle

The idea of problem chains distinguishes two views on an architecture model:

- A design artifact that represents a solution for some design problem
- A frame of reference that delimits the design or solution space.

These views are illustrated in Figure 6 (left).

In general, requirements engineering starts with some organizational change that needs to be addressed. This problem cannot be approached 'from scratch', but has to take the current organization into account, as represented by architecture model A1. This means that any goal or requirement that is defined in motivation model M should be defined 'relative' to architecture A1 in order to address the change. In addition, architecture A1 may define principles that should be adhered to by the requirements engineer or designer and constrain the possible solutions to realize the change. In this situation, architecture A1 acts as a frame of reference for requirements engineering.

Subsequently, a new architecture A2 is designed that realizes a solution for the goals and requirements in motivation model M. In this situation, architecture A2 is considered a design artifact that results from requirements engineering.

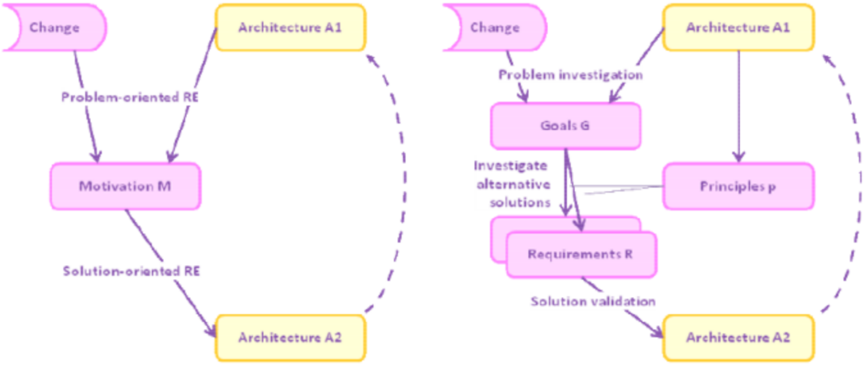


Figure 6. Architecture as a design artifact and as a frame of reference

Figure 6 (right) depicts a further decomposition of requirements engineering into three steps:

- **Problem investigation**, which focuses on the problem, i.e., the organizational change, by identifying and analyzing its cause in terms of the involved stakeholders and their concerns, and by eliciting goals to deal with the change. Goals are structured and analyzed for consistency and completeness. In addition, the impact that goals may have on each other is analyzed, e.g., to detect conflicts.

- **Investigate solution alternatives**, which refines the goals in order to find possible solutions to realize them. The impact analysis from the previous step typically triggers the identification and elaboration of alternative solutions. These solutions are guided and constrained by architecture principles, which are specialized into requirements for the specific problem and solution at hand.

- **Solution validation**, which validates alternative solutions and chooses the 'best' among them. This choice is, amongst others, influenced by the impact each solution has on the desired goals, i.e., how well the solution satisfies the goals, and by how well each solution conforms to the architecture principles that apply.

The aforementioned steps constitute a generic requirements engineering cycle (RE cycle) that can be repeated at successive phases in the development of some enterprise architecture, as indicated by the dashed arrows in figure 6.

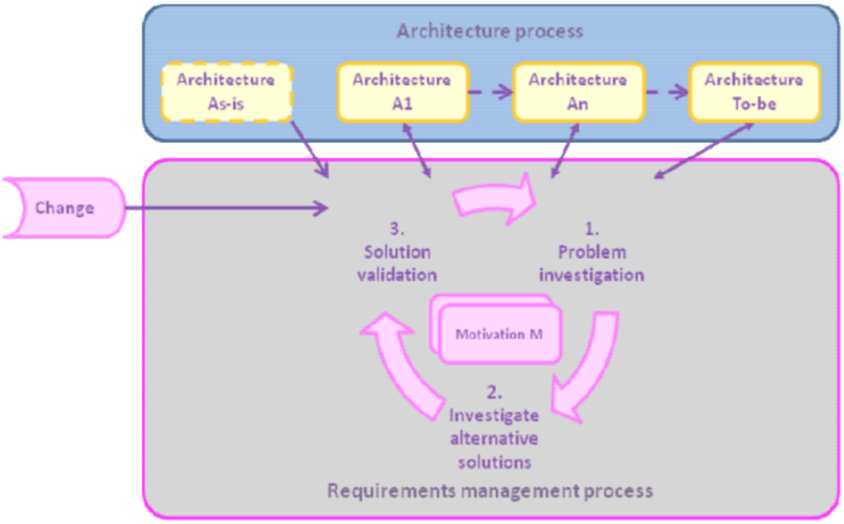


Figure 7. Application of the requirements cycle

Figure 7 illustrates the repeated application of the RE cycle in the phases of some architecture development process. The models in the center of the cycle illustrate the motivation models that are produced during the application of the engineering cycle. It is important to observe that after the elicitation of some goal in a certain cycle, this goal not necessarily has to be refined towards a solution in the same cycle. Depending on the type of goal, a goal may have to wait till the proper design phase before it is addressed. For example, goals that are related to technology may have to wait till later phases in which one decides about how enterprise applications are implemented using software and hardware.

The requirements engineering cycle plays a role in all phases of the TOGAF ADM, as will be briefly explained below.

Preliminary Phase

Business requirements management starts in the Preliminary Phase, which prepares the organization for "doing architecture". In this phase, the major stakeholders that have an interest in architecture are identified, the organizational context for working under architecture is sketched, and the high-level business goals are documented.

Also, in Preliminary Phase the architecture framework to be used is established. Possibly, the ADM and the modeling language are tailored towards specific needs of the enterprise. This may also involve tailoring of the requirements management process and, e.g., establishing the template to be used to document requirements.

Finally, during the Preliminary Phase, the architect identifies and harvests the architecture principles, based on the high-level business goals. Specific requirements are not yet defined: this is done at the start of a specific architecture development cycle, in Phase A. A method for constructing principles is out of scope for this paper. We refer to TOGAF for more information about principles development [10].

Phase A: Architecture Vision

Phase A is concerned with the preparation of a specific architecture development cycle. Possibly, additional stakeholders, their concerns and goals are defined that are specific to the architecture to be developed. Also, the relevant business goals from the Preliminary Phase are refined. Subsequently, a first set of requirements is defined on the architecture to be developed, based on which an Architecture Vision is created that presents a high-level aspirational view of the Baseline and Target Architectures. TOGAF proposes the use of so-called Business Scenarios to identify the requirements and to develop the Architecture Vision.

These activities fit very well in our proposed method for requirements management. During the ‘problem investigation’ step, the business goals are investigated, stakeholders concerns are identified, and goals are elicited to address these concerns. Based upon these goals, in the ‘investigate solution alternatives’ step, requirements are identified and assigned to the first high-level architectural vision elements. The solution alternatives investigated in this step must comply with the architecture principles identified in the Preliminary Phase.

In this phase the architecture principles may still need to be elaborated and validated against the detailed business goals. This also happens in the ‘investigate solution alternatives’ step. An important activity in this phase is to demonstrate that the chosen solution direction really satisfies the business goals and stakeholder concerns. Because we use a goal-oriented method, we can easily trace the architectural elements back to the business goals and stakeholder concerns.

Phases B, C, and D: Business, Information Systems & Technology Architecture

The objective of the ADM phases B, C and D is to elaborate the high-level Baseline and Target Architectures at the Business, Information Systems and Technology levels. This involves, amongst others, a more detailed analysis and definition of the identified requirements. One may choose to distinguish between different types of requirements, such as functional and non-functional requirements, assumptions and constraints. For validation purposes, TOGAF recommends a trade-off analysis and a gap analysis.

These activities again fit perfectly in the presented reference method. During problem investigation, the previously identified solution is investigated. Possibly, additional stakeholders and concerns are identified. Additional requirements (both functional and non-functional) are identified in the 'investigate solution alternatives' step.

Principles play an important role during these phases, because the relevant ones need refinement into requirements. During solution validation we can choose between the alternatives and select the most appropriate one. The Business Architecture, Information Systems Architecture and Technology Architecture are refined by the reference method in an iterative way. The results from the Business Architecture phase are used as a starting point for the problem investigation step of the Information Systems Architecture phase, and the results of the Information Systems Architecture phase, in turn, are used as a starting point for the problem investigation step of the Technology Architecture phase.

One of the central deliverables that TOGAF describes for these phases is the Architecture Requirements Specification, which documents the requirements that outline what an implementation project must do in order to comply with the architecture. This deliverable is complementary to the Architecture Definition Document, which provides a qualitative view of the solution and aims to communicate the intent of the architect.

Phases E and F: Opportunities and Solutions & Migration Planning

Phases E and F of the ADM are concerned with identifying and planning the projects for realizing the architecture. Therefore, these phases do not involve any actual requirements engineering activities. However, they do involve the assignment of requirements to the identified projects and work packages.

Phase E is the first phase which is directly concerned with the implementation of the Target Architecture. It identifies the programs, projects and portfolios that deliver the target architecture from the previous phases. It uses the previously identified functional requirements to identify work products and realization projects. For these realization projects, an implementation approach needs to be chosen, for example COTS selection, bespoke systems specification, etc. This does effect the requirements engineering process that is responsible for solution realization, but falls outside the scope of TOGAF.

Phase F is concerned with the planning of the migration process from the Baseline Architecture to the Target Architecture. The previously identified projects are prioritized and budgets assigned. The results from this phase may result in changes to the Architecture Definition Document and corresponding Architecture Requirements Specification.

We make a distinction between architectural requirements and realization requirements. Realization requirements are the requirements identified in solution realization projects. They refine the architectural requirements, but new requirements may arise as well, depending on the chosen solution type.

Phase G: Implementation Governance

During Phase G of the ADM, the actual solutions are realized and monitored to ensure compliance with the architecture. The previously identified projects are fed with the architectural requirements defined so far. Basically, this jump-starts the classical requirements engineering process. Classical requirements engineering is concerned with specifying the requirements for a solution (e.g. new applications). Requirements management in phase G is concerned with validating these detailed requirements and solution specifications to the architectural context and requirements.

This fits the presented reference method. The architectural requirements form the starting point of the problem investigation step, in which additional stakeholders may be identified including concerns and goals related to the implementation of the architecture. Next, detailed requirements are elicited to investigate alternative implementation solutions. And finally, these solutions are validated against the architectural requirements, and a choice is made among the alternatives.

Phase H: Architecture Change Management

Phase H of the ADM is concerned with managing changes in the business environment and their impact on the enterprise architecture. The drivers for change may be manifold and at different levels, e.g., change of business strategy, new technology or customer demands.

The drivers for change are input for the requirements engineering cycle. First, the drivers and related stakeholder concerns are analyzed in the problem investigation step, which may lead to new or adapted stakeholder goals. Second, alternative business requirements are derived to realize these goals and checked for conformance with existing requirements and principles. In the third step, the impact of the requirements is analyzed to assess whether the change can be handled via normal change management techniques or a (partial) redesign of the architecture is needed. The latter requires a new iteration of the ADM cycle.

Motivation concepts – Way of modeling

The core concepts of ArchiMate focus on what could be called the "extensional" aspects of the enterprise – i.e., its appearance as an operational entity. The "intentional" aspects – i.e., the business goals, principles and requirements that *motivate* the design of the enterprise – are not covered by the core concepts.

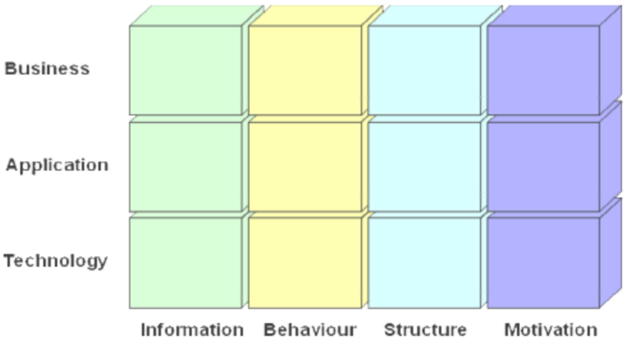
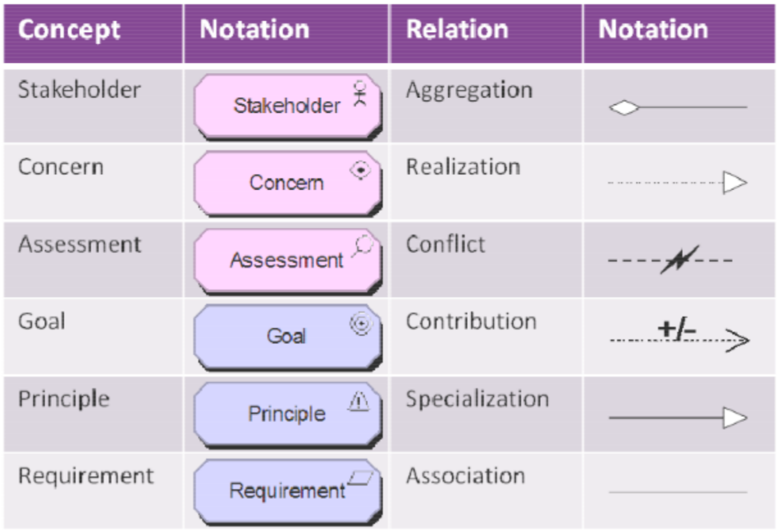


Figure 8. Extension of the ArchiMate framework

To support the modeling of motivational properties, we propose to extend the ArchiMate 1.0 framework with the 'motivation' aspect. Figure 8 depicts this extension. The motivation aspect is concerned with the intentions – goals, principles and requirements – of the enterprise, corresponding to the motivation ('why') column of the Zachman framework [12, 13].

Intentions are pursued by people, the *stakeholders*, which have certain areas of interest, called *concerns*. These concerns are used to organize the stakeholders' intentions. *Assessments* of the concerns are needed to decide whether existing intentions need to be adjusted or not. The actual intentions are represented by *goals*, *principles* and *requirements*.

The following table depicts the concepts and relationships that are used to model the motivation aspect of enterprise architectures.



Concept	Notation	Relation	Notation
Stakeholder	Stakeholder	Aggregation	◇———
Concern	Concern	Realization	- - - - ▷
Assessment	Assessment	Conflict	- - - -↯- - -
Goal	Goal	Contribution	- - - +/- - - ->
Principle	Principle	Specialization	———▷
Requirement	Requirement	Association	———

A **stakeholder** represents an individual, team, or organization with an interest in the outcome of the architecture. This definition is adopted from TOGAF [10]. Examples of stakeholders are the board of directors, shareholders, customers, business and application architects, legislative authorities.

A **concern** represents some key interest that is important to certain stakeholders in a system, and determines the acceptability of the system. A concern may pertain to any aspect of the system's functioning, development, or operation, including non-functional considerations such as performance and security. This definition is also adopted from TOGAF [10].

An **assessment** represents the outcome of the analysis of some concern, revealing the strengths, weaknesses, opportunities or threats that may trigger a change to the enterprise architecture. These are addressed by the definition of new or adapted business goals.

The **association relationship** of ArchiMate is reused to relate stakeholders to concerns and concerns to assessments. A stakeholder may have multiple concerns, and a concern may be shared by multiple stakeholders. An assessment typically assesses a single concern, but could involve multiple concerns. A concern may be analyzed through multiple assessments.

Figure 9 depicts part of the stakeholders view of PRO-FIT, a company that sells insurances. This view shows two stakeholders: the 'Board' and the 'Customer' of PRO-FIT. Each stakeholder has a number of concerns, which may be shared, e.g., ‘Customer satisfaction’. Analysis of this leads to the assessments ‘Complaining customers’ and ‘Leaving customers’. The former assessment is decomposed into sub-assessments to describe, e.g., categories of complaints in more detail. Also, stakeholders and concerns can be decomposed. For example, ‘Profit’ is modeled as a sub-concern of ‘Stockholder satisfaction’.

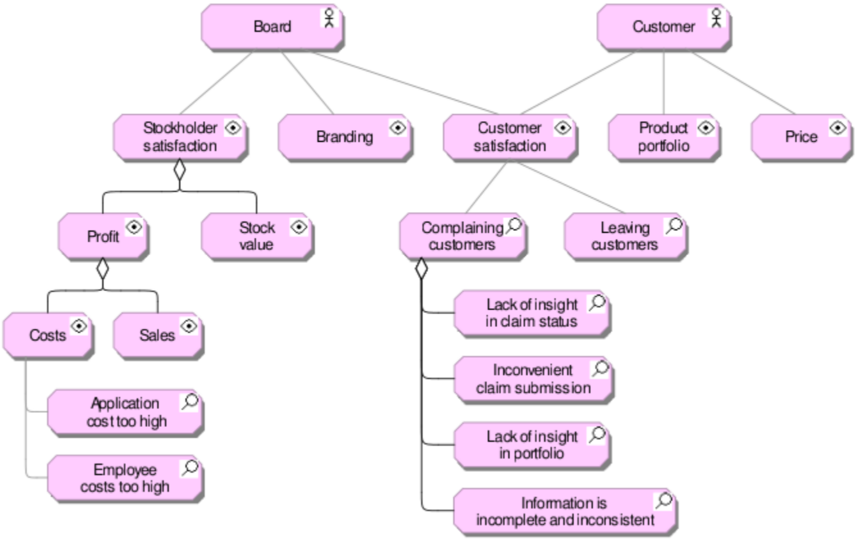


Figure 9. Stakeholder view of PRO-FIT

A **goal** represents some end that a stakeholder wants to achieve. In principle, an ‘end’ can represent anything a stakeholder may desire, such as a state of affairs, a produced value or a realized effect. Examples of goals are: to increase profit, to reduce waiting times at the helpdesk or to introduce on-line portfolio management.

In general, one may distinguish between goals that are precise and measurable, so-called ‘hard’ goals, and goals that merely indicate some direction, so-called ‘soft’ goals. Typically, soft goals are more abstract and need to be decomposed into hard goals during the design process. For example, Figure 11 depicts the goal ‘reduce workload employees’, which addresses the assessment “Employee costs too high”. This goal can be made measurable by using ‘workload’ as an indicator and specifying a target, e.g., ‘reduce workload by 10%’. Alternatively, ‘reduce workload’ can be decomposed into the sub-goals ‘reduce manual work’ and ‘reduce interaction with customer’. However, the corresponding goals are still soft. Similar to the super-goal, both sub-goals can be made measurable by adding a target or by further decomposition. Figure 10 illustrates this example.

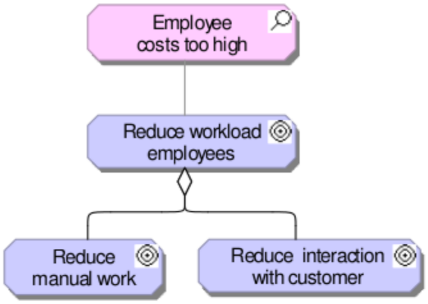


Figure 10. Example of a goal and sub-goals that address an assessment

A **requirement** represents a desired property that must be realized by a system. The term 'system' is used in a broad sense, i.e., as a group of (functionally) related elements, each of which may be considered as a system again. A system may refer to any structural, behavioral or informational element of some organization, such as a business actor, application component, business process, application service, business object, data object, etc.

Requirements model the properties of these elements that are needed to achieve the 'ends' that are modeled by the goals. In this respect, requirements represent the 'means' to realize goals. During the design process, goals may be decomposed until the resulting sub-goals are sufficiently detailed to enable their realization by properties that can be exhibited by systems. At this point, goals can be realized by requirements that assign these properties to the systems.

For example, one may identify two alternative requirements to realize the goal to improve portfolio management (see Figure 11):

- By assigning a personal assistant to each customer.
- By introducing on-line portfolio management.

The former requirement will be realized by a human actor and the latter by a software application. These requirements may be decomposed further to define the requirements on the human actor and the software application in more detail.

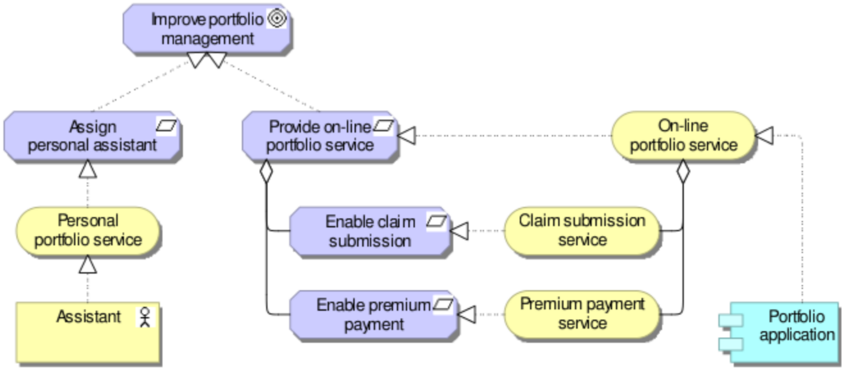

Improve portfolio management
Assign personal assistant
Provide on-line portfolio service
On-line portfolio service
Personal portfolio service
Enable claim submission
Claim submission service
Assistant
Enable premium payment
Premium payment service
Portfolio application

Figure 11. Example of alternative realizations of a goal

A **principle** represents a general desired property that guides the design and evolution of systems in a given context. Principles are strongly related to goals and requirements. Similar to requirements, principles define desired properties of systems. However, principles are broader in scope and more abstract than requirements. Whereas a principle defines a *general* property that applies to any system in a certain context, a requirement defines a property that applies to a *specific* system.

A principle needs to be specialized into requirements in order to enforce that some system conforms to the principle. For example, the principle “Systems should be customer facing” can be specialized into the requirements “Provide on-line portfolio service” and “Provide on-line information service” in the given problem context of the organization. This is illustrated in Figure 12.

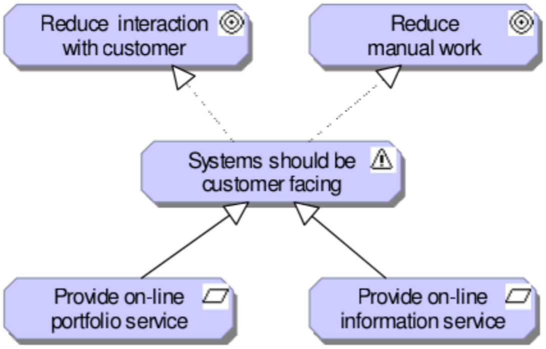

Reduce interaction with customer
Reduce manual work
Systems should be customer facing
Provide on-line portfolio service
Provide on-line information service

Figure 12. Example of a principle that is specialized into requirements and motivated by goals

A principle is motivated by some goal. For example, the aforementioned principle may be motivated by the goals to reduce interaction with customers and to reduce manual work.

The **aggregation relationship** models the decomposition of some intention, i.e., a goal, requirement or principle, into more fine-grained intentions.

The **realization relationship** models that some end is realized by some means. Realization is used to describe how

- A goal (the end) is realized by a principle or requirement (the means);
- A requirement (the end) is realized by a system (the means), which can be represented by a passive structure element, a behavior element or a passive structure element.

The **conflict relationship** models that the realization of two intentions mutually exclude each other. It is used to describe that two intentions cannot be realized both, and as such are in conflict with each other. For example, as depicted in Figure 14, the goal to reduce interaction with the customer conflicts with the requirement to assign a personal assistant to each customer.

The **contribution relationship** models that the realization of some intention contributes positively or negatively to the realization of another intention. For example, Figure 13 depicts the contribution of the alternative requirements to realize the goal “Improve portfolio management” to the goal “Increase customer satisfaction” and the principle “Systems should be customer facing”. An attribute can be used to indicate the direction and strength of the contribution, e.g., using numeric values or a scale such as {++, +,   , --}. The modeling of contribution relationships can be used in a trade-off analysis to choose between both alternatives.

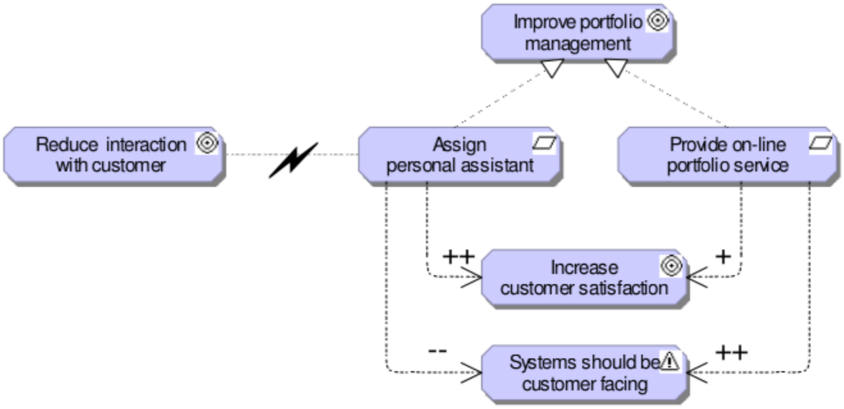

Improve portfolio management
Reduce interaction with customer
Assign personal assistant
Provide on-line portfolio service
++
+
Increase customer satisfaction
--
++
Systems should be customer facing

Figure 13. Example of contribution and conflict relationships

A positive contribution relationship and negative contribution relationship are weaker than the realization and conflict relationships, respectively.

ArchiMate's **specialization relationship** indicates that an object is a specialization of another object. In the current context, this relationship is used in particular to describe that a principle is specialized into a requirement. This has been illustrated in Figure 13.

Techniques and tools

A requirements model may grow quickly in subsequent RE cycles. Furthermore, the relationships involving some goal, principle or requirement may be distributed over different views of the model. Therefore, tool support to create, manipulate and analyze requirements models is indispensible.

Model creation and manipulation requires an editing tool that supports both the ArchiMate core concepts and the motivation extension proposed in this paper.

Automated techniques are required for the analysis of motivation models. In order to deal with complexity, the analysis of a motivation model may be separated into the analysis of distinct aspects, each of which stands for a particular viewpoint: e.g., the goals and concerns per stakeholder, or the conflict and contribution relationships for each set of goals. Automated tool support is needed to generate and analyze views. Useful categories of analysis include: completeness, consistency, trade-offs between alternative solutions, and impact of change. These types of analysis build on the *traceability* between model elements, which is obtained by using the method and language for requirements management as described in the previous sections.

Views

Several enterprise architecture tools support the generation of views from a single underlying model. The list below gives some suggestions for useful viewpoints with concepts from the motivation extension:

- *Stakeholders viewpoint*, which focuses on modeling the stakeholders, their concerns, the assessments of these concerns, and the initial goals to address these concerns and assessments;
- *Goal refinement viewpoint*, which focuses on refining the initial, high-level goals into more concrete (sub-)goals using the decomposition relation, and finally into requirements using the means-end relation.
- *Goal contribution viewpoint*, which focuses on modelling and analyzing the contribution and conflict relations between goals (and requirements).
- *Principles viewpoint*, which focuses on modeling the relevant principles and the goals that motivate these principles;
- *Requirements realization viewpoint*, which focuses on modeling the realization of requirements by means of core elements, such as actors, services, processes, application components, etc.

Completeness

Completeness of a requirements model means that each goal, principle or requirement is traceable from its origin to its realization(s). E.g., it should be possible to trace a requirement *backwards* to one or more stakeholders via the goals they realize and the concerns and assessments that are addressed by these goals. Also, it should be possible to trace a requirement *forwards* to the behavioral and structural elements that realize the requirement.

Consistency

Various techniques can be used to analyze the consistency of a requirements model. Examples are: the detection of cyclic dependencies and conflicts between goals or the detection of syntactically and/or semantically equivalent goals and requirements.

Trade-off between alternative solutions

During goal refinement alternative solutions may be identified. Contribution relationships can be used to model how well the alternatives contribute to certain goals that have been selected as decision criteria. Figure 14 depicts an example of two alternatives that contribute to two decision criteria.

Impact of change

Traceability enables impact of change analysis, which plays an important role in Phase H of the ADM. For example, in case of changing legislation, certain high-level goals that address the compliance to this legislation may change. By tracing the relationships from these goals to the requirements and the architectural elements that realize these requirements, one can determine which business and IT elements are affected by this change. Furthermore, impact of change can be traced in the 'backwards direction'. For example, when some IT requirement cannot be met, the business process that imposes this requirement cannot be realized. As a consequence, the organization may not be able to offer some of its business services. Following these dependencies further, one may assess the impact of the IT requirement on the business goals and strategy of the organization.

Conclusions

In this paper we have presented an extension to TOGAF and ArchiMate supporting requirements management and principles. This extension constitutes a practical way of *thinking*, *working*, *modeling* and *supporting*.

We have demonstrated how enterprise architecture design, requirements management and the application of principles can be related through the RE cycle. This contributes to a better grip on the architecture development process, consisting of:

- *Better motivation and foundation* of architecture models and design choices;
- *Better insight* in architecture models due to enhanced traceability between motivation elements and architecture elements. This also enables *clearer communication* of architecture models;
- Additional and *better analysis techniques* for consistency, completeness, trade-off between alternative solutions and impact of change;
- A *generic method and modeling language* that can be applied to enterprise architecture design, but might also be applicable to business process design and functional design.

The integration of enterprise architecture design, requirements management and the application of principles also contributes to the ‘closing’ of the Business-IT gap. Architecture models are aligned to the vision, mission and strategy of the organization by eliciting goals from stakeholder concerns and the organization’s business plan and by refining these goals via principles and requirements into products, business services, business processes and software applications that realize these goals.

References

[1] A.I. Antón, W.M. McCracken, and C. Potts. Goal decomposition and scenario analysis in business process reengineering. *Proceedings of the 6th international conference on Advanced information systems engineering table of contents*, pages 94–104, 1994.

[2] F.P. Brooks. No silver bullet: Essence and accidents of software engineering. *IEEE computer*, 20(4):10–19, 1987.

[3] A. Dardenne, A. van Lamsweerde, and S. Fickas. Goal-directed requirements acquisition. *Science of Computer Programming*, 20(1-2):3–50, 1993.

[4] D. Firesmith. Prioritizing Requirements. *Journal of Object Technology*, 3(8):35–47, 2004.

[5] J.A. Goguen and C. Linde. Techniques for Requirements Elicitation. *Requirements Engineering*, 93:152–164, 1993.

[6] The Open Group. *ArchiMate 1.0 Specification*. Van Haren Publishing, 2009.

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

[8] H. Jonkers, E. Proper, and M. Turner. TOGAF 9 and ArchiMate 1.0. Submitted for review at the The Open Group.

[9] D.A.C. Quartel, W. Engelsman, H. Jonkers, and M.J. van Sinderen. A goal-oriented requirements modeling language for enterprise architecture. in *Proceedings of the 13th IEEE International EDOC Enterprise Computing Conference,* Auckland, New-Zealand, Sept. 2009, pp. 3-13.

[10] The Open Group. TOGAF™ Version 9. Van Haren Publishing, 2009..

[11] A. van Lamsweerde et al. Goal-Oriented Requirements Engineering: A Guided Tour. *Proceedings of the 5th IEEE International Symposium on Requirements Engineering*, page 249, 2001.

[12] J.A. Zachman. Enterprise Architecture: The Issue of the Century. *Database Programming and Design*, 10:44–53, 1997.

[13] J.A. Zachman. A Framework for Information Systems Architecture. *IBM Systems Journal*, 38(2/3):454–470, 1999.

About the Authors

Dick Quartel is a Senior Researcher at Novay. In this capacity, he is involved in research, consultancy, project management and project development. The mission of Novay is to facilitate innovation enabled by ICT in companies and public organizations. Before joining Novay, Dick worked as an assistant professor at the University of Twente. His expertise and interests include enterprise and service-oriented architecture, business process (re)design, model-driven development of services and software, service composition and interoperability, and requirements modeling. Dick has worked in several national and European projects.

Wilco Engelsman is a Research Consultant at BiZZdesign. In this capacity, he is involved in the company's new developments in the areas of requirements engineering and enterprise architecture; he participates in multi-party research projects, as well as in consultancy for customers.

Henk Jonkers is a Senior Research Consultant at BiZZdesign. In this capacity, he is involved in the company's new developments in the areas of business process engineering and enterprise architecture, he participates in multi-party research projects, as well as in consultancy for customers. Henk was one of the main developers of ArchiMate and an author of the ArchiMate 1.0 Specification. He is TOGAF 8 and 9 certified.

About The Open Group

The Open Group is a vendor-neutral and technology-neutral consortium, whose vision of Boundaryless Information Flow™ will enable access to integrated information within and between enterprises based on open standards and global interoperability. The Open Group works with customers, suppliers, consortia, and other standards bodies. Its role is to capture, understand, and address current and emerging requirements, establish policies, and share best practices; to facilitate interoperability, develop consensus, and evolve and integrate specifications and Open Source technologies; to offer a comprehensive set of services to enhance the operational efficiency of consortia; and to operate the industry's premier certification service, including UNIX® system certification. Further information on The Open Group can be found at www.opengroup.org.