# ArchiMate® for Integrated Modelling Throughout the Architecture Development and Implementation Cycle

Henk Jonkers
BiZZdesign
Enschede, the Netherlands
h.jonkers@bizzdesign.nl

Marc M. Lankhorst, Dick A.C. Quartel
Novay
Enschede, the Netherlands
marc.lankhorst@novay.nl
dick.quartel@novay.nl

Erik (H.A.) Proper
Public Research Centre Henri Tudor
Luxembourg-Kirchberg, Luxembourg
and
Radboud University Nijmegen
Nijmegen, the Netherlands
erik.proper@tudor.lu

Maria-Eugenia Iacob
University of Twente
Enschede, the Netherlands
m.e.iacob@utwente.nl

*Abstract*—**The ArchiMate standard offers an integrated language for enterprise architecture modelling. It allows for the description and visualization of different architecture domains, as well as their underlying relationships and dependencies. Since its adoption as a standard of The Open Group, the international interest in ArchiMate has been growing rapidly. ArchiMate complements TOGAF, the standard of The Open Group for developing enterprise architectures. To provide modelling support throughout TOGAF's architecture development and implementation cycle as defined by TOGAF, two extensions to the original ArchiMate language have been proposed: a *Motivation* extension and an *Implementation and Migration* extension.**

***Keywords – enterprise architecture; modelling; ArchiMate; TOGAF;***

## I. INTRODUCTION

The ArchiMate language for enterprise architecture modelling language has been developed with the aim to provide a uniform representation for enterprise architecture (EA) descriptions [5][7]. It offers an integrated architectural approach by which organizations can describe and visualize different architecture domains, as well as their underlying relationships and dependencies.

Within larger organizations one can typically find various architecture domains, such as: organizational structures, products, business processes, information systems, applications and technological infrastructure. Traditionally, each architecture domain employs specific models and visualizations, which simplifies communication, discussion and analysis *within* the domain. However, the relations between these different domains are in many cases unclear. Moreover, these domains tend to (at least partially) overlap. Therefore, ArchiMate provides a unified way to model enterprise architectures, while integrating the various domains and describing them in an easily readable way, as illustrated in Figure 1. In addition, a distinction is made between a business layer, an application layer, and a layer with the underlying (IT) technological infrastructure.

The concept of service plays a central role in ArchiMate [5][7]. Services are used to "bind" together the layers (applications provide services to business processes, while applications on heir turn use infrastructure services). Furthermore, within a layer services can be used as well to encapsulate behaviour. This enables the use of a services oriented architecture (SoA) style from business processes, via applications to the underlying infrastructures.

Since ArchiMate is positioned at the level of enterprise architecture, this also implies that the ArchiMate language does not provide the level of detail one would typically find in languages used at the "design level" [11]. For example, while ArchiMate features concepts such as business event and junction, it does not provide the rich detailed set of gateways, et cetera as offered by a language such as BPMN [10]. Similarly, in contrast to languages such as UML [9], it does not provide concepts to model the details of software applications. At the same

978-0-7695-4535-6/11 $26.00 © 2011 IEEE
DOI 10.1109/CEC.2011.52

time, refinement/abstraction mechanisms can be used to maintain the connection between, e.g., a BPMN or UML model and an ArchiMate model [5].

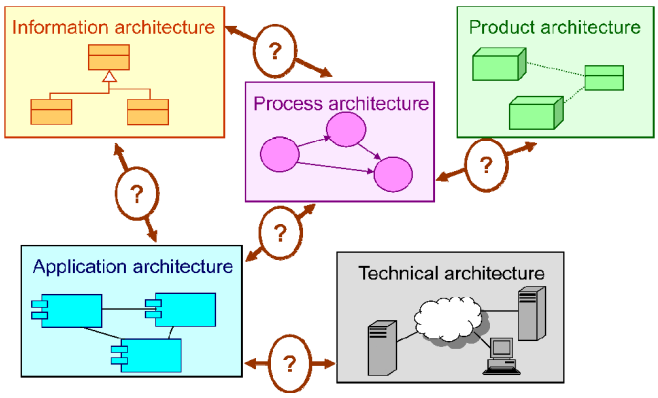


Figure 1. Integration of architectural domains

There are several content frameworks or metamodels that define the types of "building blocks" that are relevant in an enterprise architecture, e.g., the Integrated Architecture Framework [3], TOGAF's Content Metamodel [8], or defence frameworks such as DoDAF and MODAF. While we have drawn inspiration from these frameworks, the ArchiMate language has been designed in a more structured way, by defining a generic structure that is made specific for the different architectural layers (as will be explained in Section III). Also, ArchiMate has a limited set of relation types that are used throughout the metamodel. Finally, unlike these other frameworks, ArchiMate provides a standard graphical notation for the modelling concepts and relations (in this respect, it is more similar to the detailed design languages such as BPMN and UML).

## II. An Integrated Approach to Enterprise Architecture

Frameworks for enterprise architecture vary in the types of support that they offer. They may have, among others, any combination of the following ingredients:

- A process ("*way of working*" [12]) for creating architectures; this may be accompanied by guidelines, techniques and best practices.
- A set or classification of viewpoints.
- A language for describing architectures (defining concepts and relationships, but also a notation).
- The concept of a (virtual) architecture repository, possibly containing predefined architectural artefacts and (reference) models.

The core of TOGAF is formed by its process, the Architecture Development Method (ADM). It also includes the identification of viewpoints, techniques and reference models. However, it does not define an actual modelling language. The TOGAF Architecture Content Framework does indeed identify relevant architecture building blocks, but it does not constitute a precisely defined language, nor does it provide a notation for these building blocks. ArchiMate complements this by defining a fully worked out (graphical) modelling language, including the definition of relevant viewpoints. This language also provides a concrete visualization of the views identified in TOGAF.

TOGAF and ArchiMate share their view on the use of viewpoints, and the concept of an underlying common repository of architectural artefacts and models; i.e., they have a firm common foundation. However, TOGAF and ArchiMate complement each other with respect to the definition of an architecture development process and the definition of an enterprise architecture modelling language. Together, they make up a complete, integrated approach for delivering enterprise architecture.

## III. Structure of the ArchiMate Language

In this section we briefly discuss the core structures of the ArchiMate language. In [11] a more detailed account is provided of the requirements on the language, and the design decisions that underpin its design.

### A. Core Concepts

To arrive at a language that is easy to learn and understand, a conscious decision was made to limit the set of core modelling. Therefore, a small number of generic modelling concepts have been created that essentially re-appear (in different variations) on the various layers of the language. First, we distinguish between the *structural* or *static* aspect and the *behavioural* or *dynamic* aspect. Behavioural concepts are assigned to structural concepts, to show who or what displays the behaviour. In addition to

*active* structural elements (the business actors, application components and devices that display actual behaviour, i.e., the 'subjects' of activity), we also recognize *passive* structural elements, i.e., the *objects* on which behaviour is performed.

Second, we make a distinction between an *external view* and an *internal* view on systems. When looking at the behavioural aspect, these views reflect the principles of service orientation as introduced in the previous section. The *service* concept represents a unit of essential functionality that a system exposes to its environment. For the external users, only this external functionality, together with non-functional aspects such as the quality of service, costs etc., are relevant. Services are accessible through *interfaces*, which constitute the external view on the structural aspect.

Figure 2 summarizes the resulting generic core concepts of the language, as well as their main relationships

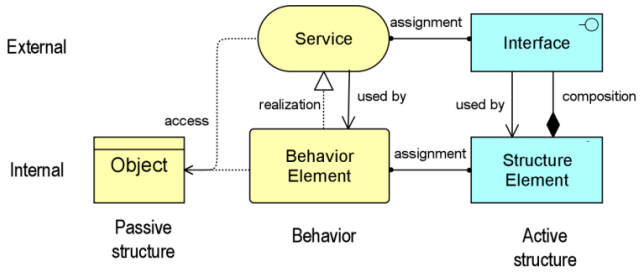


Figure 2. Core Concepts of the ArchiMate Language

### B. *Services as a Linking Pin Between Layers*

In ArchiMate, the concept of service is defined as *the externally observable behaviour of a system*[1] *that may have some added value for that system's environment*. It is therefore natural to expect that, in the case of architecture layers, higher layers use the services supplied by the lower layers, since higher layers can be seen as the "environment" of the lower layers (see Figure 3). The enterprise's environment is the "end-user" of the services offered by the business layer of the enterprise. The business layer makes use of the services exposed by application layer (e.g., in order to support and automate its business processes). The application layer uses the services supplied by the technology layer (e.g., to make use of the physical resources – servers, networks etc. - in order to run its applications).

[1] System in the general sense, and not just as a synonym to application

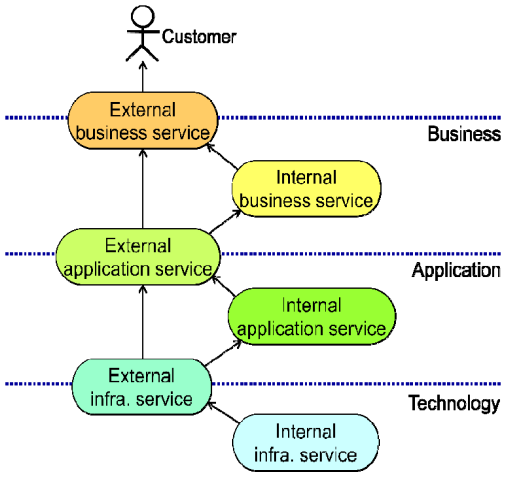


Figure 3. Services

In addition to the types of services mentioned above, the ArchiMate language distinguishes within each layer between internal and external services. Internal services are the services (added values) supplied to entities within the same layer. External services are the services made available to entities from outside that layer.

## IV. Creating Architecture Models with ArchiMate

The primary use of ArchiMate in the context of TOGAF will be the representation of architecture models. TOGAF distinguishes four architectures: the Business Architecture (created in Phase B of the ADM), the Application Architecture and Data Architecture (both part of the Information Systems Architecture, Phase C) and the Technology Architecture (Phase D). In all of these phases, baseline ("*as is*") and target ("*to be*") architectures are created. In Phase A (Architecture Vision) of the ADM, first global versions of these architectures are already sketched; for this, ArchiMate may also be suitable.

We illustrate the different architectures with a small example based on a fictitious insurance company. ArchiSurance is a merger of three previously independent companies: Home & Away for home and travel insurances, PRO-FIT for car insurances, and Legally Yours for legal aid insurances. The new company has a single Front Office and three separate Back Offices. ArchiSurance intends to rationalize their application portfolio, by integrating legacy applications with similar functionality from the old companies that are still in use. Note that these examples give an impression of the ArchiMate lan-

guage, but do not show all the concepts. For a complete overview of the language, please refer to [7].

### A. Business Architecture

The Business Architecture provides the context for system development trajectories, showing, among others, the main business processes, the actors (or roles) performing these processes, and the information (objects) exchanged between the processes.

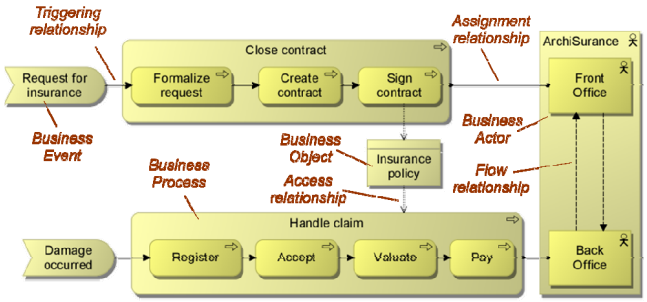


Figure 4. Baseline and Target Business Architecture

Figure 4 shows an example of a Business Architecture expressed in ArchiMate. We assume that the business architecture of ArchiSurance does not change in the application rationalization process.

### B. Application Architecture

The Application Architecture shows the applications or application components, their relationships and their functionality. Figure 5 shows the baseline Application Architecture of ArchiSurance. The functionality that the applications offer to their environment is modelled with services. The service concept plays a central role in ArchiMate, also in the Business Architecture and the Technology Architecture (although this is not shown in our example), and in particular as a linking pin between the different architectures.

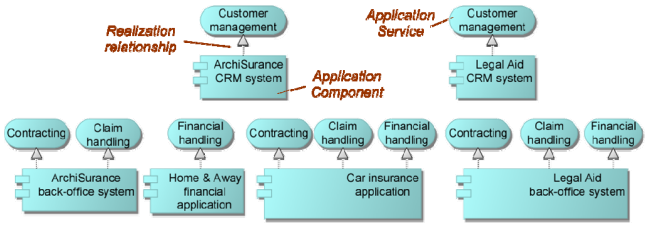


Figure 5. Baseline Application Architecture

Figure 6 shows the target Application Architecture of ArchiSurance, in which the legacy applications have been replaced by a single back-office system and a single CRM system for the whole company.

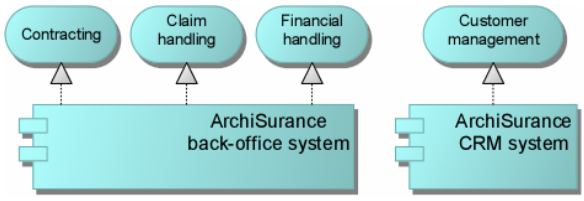


Figure 6. Target Application Architecture

In ArchiMate, separate views can be used to show the relationships between the different architectures. As an example of this, Figure 7 shows how the services from the Application Architecture are used in the processes of the Business Architecture.

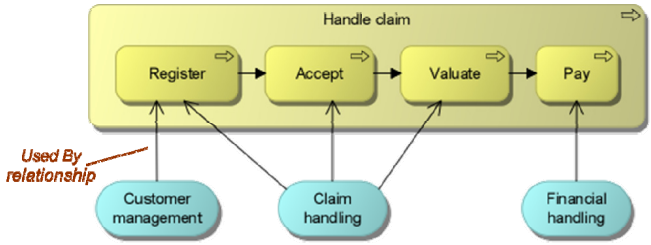


Figure 7. Business-Application Alignment (Target)

### C. Data Architecture

The Data Architecture shows the main data object used within the applications, as well as their relationships. Figure 8 shows the Data Architecture of ArchiSurance, which we assume will not change in the application rationalization process.

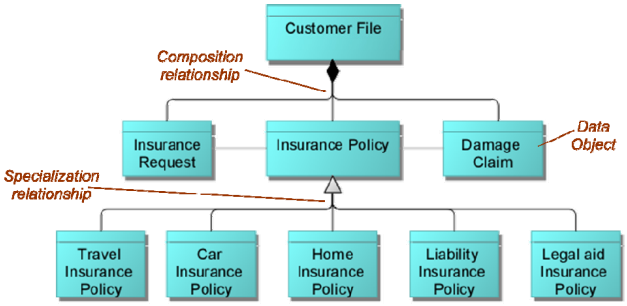


Figure 8. Baseline and Target Data Architecture

### D. Technology Architecture

The Technology Architecture shows, among others, the devices and system software on which applications run, the networks connecting devices, and artefacts that form the physical implementation of application components or data objects. Figure 9

shows the baseline Technology Architecture of ArchiSurance. There are separate application servers for the different back-office applications.

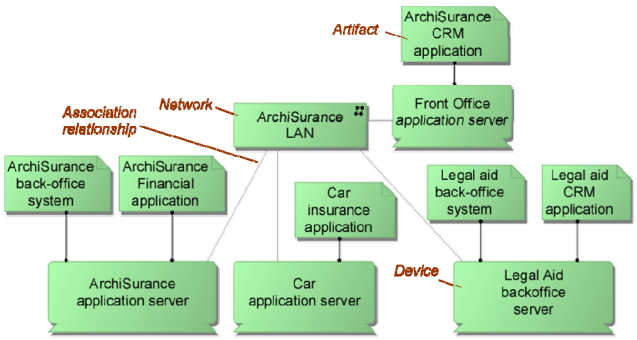


Figure 9. Baseline Technology Architecture

In the target Technology Architecture, as shown in Figure 10, some of these application servers become redundant. However, to increase reliability and availability, an additional backup server is introduced.

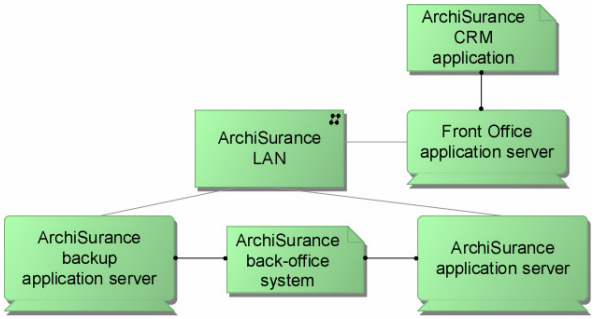


Figure 10. Target Technology Architecture

### E. *Gap Analysis*

An important step in Phases B, C and D of the TOGAF ADM is a gap analysis, which reviews the differences between the baseline and target architecture. It shows which building blocks are carried over from baseline to target, which building blocks are new in the target architecture (which can be used as a basis to decide whether to buy or build these building block), and which elements have been eliminated from the baseline architecture (on purpose or accidentally; i.e., a gap analysis can also be used as a mechanism for validation of the target architecture). Phases E, F and G of the TOGAF ADM then deal with the implementation of the proposed target architecture.

TOGAF suggests the use of a gap matrix as a technique for gap analysis. However, ArchiMate models also form a useful starting point for gap analysis, and the results can also be presented as an ArchiMate view. Figure 11 shows an example of this for the Technology Architecture.

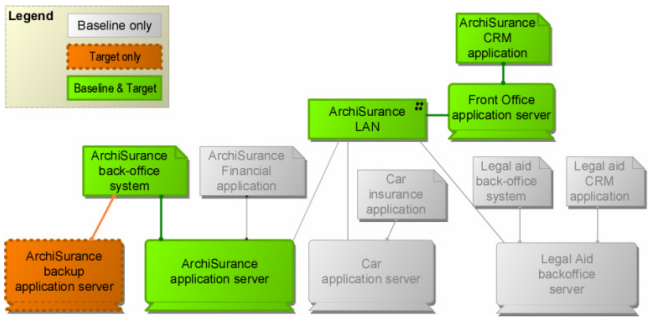


Figure 11. Technology Architecture Gap Analysis

## V. EXTENDING ARCHIMATE'S TOGAF COVERAGE

As described in the previous sections, ArchiMate version 1.0 chiefly supports modelling of the architectures in Phases B, C and D in the TOGAF ADM, as is illustrated in Figure 12. The resulting models are used as input for the subsequent ADM phases. However, modelling concepts specifically aimed at the other phases – e.g., concepts for modelling principles, goals and requirements, or concepts to support migration planning – are still missing in the language. This observation points in a direction for language extensions of ArchiMate. Currently, a proposal for ArchiMate version 2.0 is under review at The Open Group, which provides two extensions: for describing motivation (e.g. stakeholders, concerns, requirements) and for implementation and migration planning. The next subsections outline these two extensions.

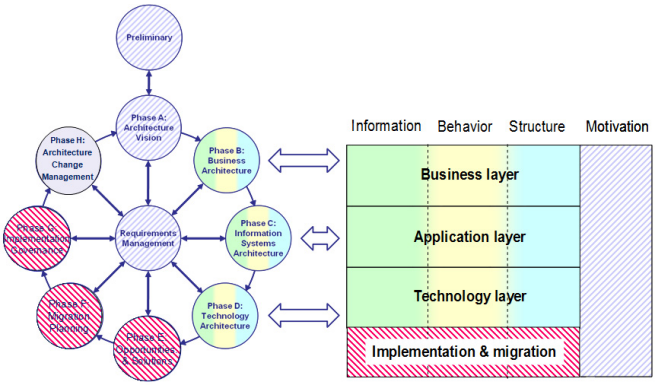


Figure 12. TOGAF ADM and ArchiMate

### A. *Motivation Extension*

ArchiMate 1.0 does not include concepts for describing the reasoning behind the various architec-

tural decisions. The proposed motivation extension [1] adds such concepts for business requirements management. They can be used the identification, description, analysis and validation of requirements at business level and their realization in enterprise architecture models as described with the current ArchiMate concepts.

The proposed motivational concepts, based on sources such as OMG's business motivation model [13], architecture principles [14], [15] and goal driven requirements engineering [16], [17], [18] are used to model the motivations, or intentions, that underlie the design of an enterprise architecture. These intentions influence, guide and constrain the design. Intentions are pursued by stakeholders, which can be individuals or groups such as a project team, enterprise or society. In addition, intentions may be organized into certain areas of interest, called concerns, such as customer satisfaction, compliance to legislation or profitability. Assessments of these concerns are needed to decide whether existing intentions need to be adjusted or not.

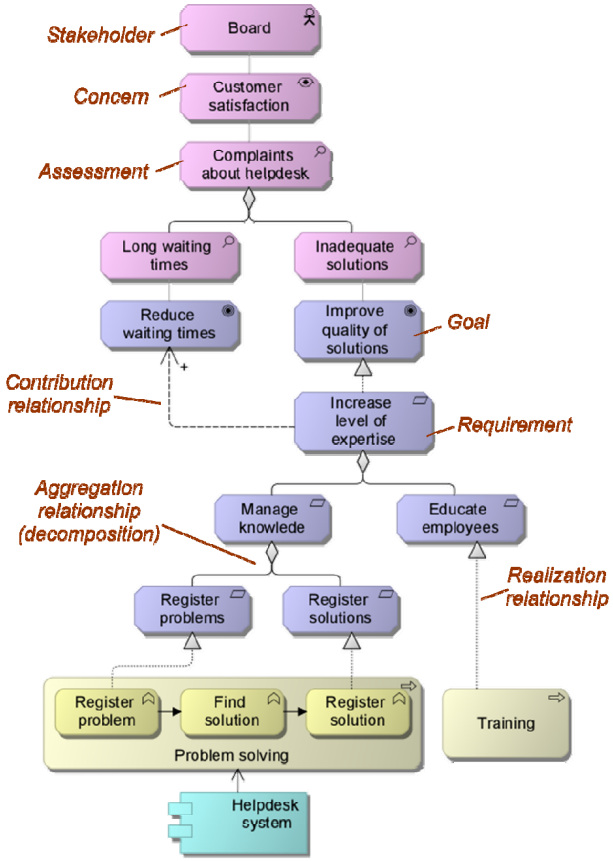


Figure 13. Stakeholders, Goals, Requirements and Architecture Artifacts

The actual intentions are represented by goals, principles and requirements. Goals represent some desired result – or end – that a stakeholder wants to achieve; e.g., increasing customer satisfaction with 10 percent. Principles and requirements represent desired properties of solutions – or means – to realize the goals. Principles represent desired properties that are required from all possible solutions in a given context; requirements represent desired properties of specific, individual solutions. For example, the requirement "*Use a single CRM system*" is a specialization of the principle "*Data should be stored only once*" by applying it to the current organization's architecture in the context of the management of customer data. The top side of Figure 13 shows the relationship between stakeholders, concerns, assessments, goals and requirements; the bottom side shows the relationship with the architecture artifacts that should realize these requirements.

### B. *Implementation & Migration Extension*

The Implementation and Migration Extension proposes several additional concepts that make possible the modelling of the architecture change process and increase the insight into these changes as well as their manageability in terms of portfolio and project management and the decision making. By defining concepts such as program, project, activity, result, gap, and plateau it is possible to connect ArchiMate with program and project management standards and best practices, such as MSP [19], PRINCE2 [21] and PMBoK [20]. The central behavioural concept in the implementation and migration extension is a project. A project is basically a management environment that has a clearly defined beginning and end date, and aims to deliver a well-defined set of goals or results.

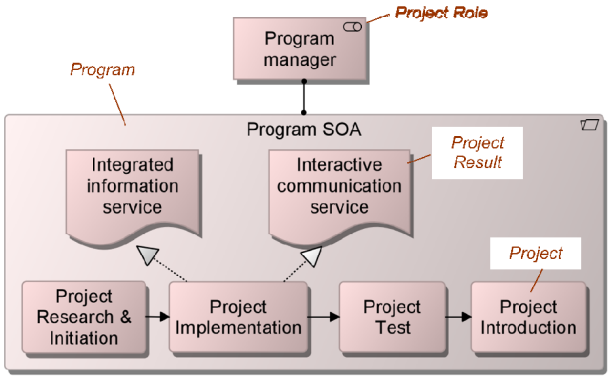


Figure 14. Programs, Projects, Project Roles and Project Results

Figure 14 shows an example of the use of projects and related concepts. A project may be subdivided into a hierarchy of project activities. A project activity may also be further subdivided in finer-grained project activities. Multiple projects which are managed together coherently, and which all contribute to a common outcome, can be grouped into a program. A program may also contain subprograms. Projects and project activities produce project results (or deliverables). These may be results of any kind, e.g., reports, papers, services, software, physical products, etc. A project result may also be (a part of) an architecture, or a solution that implements (a part of) an architecture. To each program, project or project activity, one or more project roles can be assigned. Project roles may be fulfilled by specific project actors. A single actor may be assigned to multiple roles, although there may be some restrictions on the roles that may be combined.

An important premise in TOGAF is that the various architectures are described for different stages in time. In each of the Phases B, C, and D of the ADM, a Baseline Architecture and Target Architecture are created, describing the current situation and the desired future situation. In Phase E, "*Opportunities and Solutions*", Transition Architectures are defined, showing the enterprise at incremental states reflecting periods of transition between the Baseline and Target Architectures. Transition Architectures are used to allow for individual work packages and projects to be grouped into managed portfolios and programs, illustrating the business value at each stage. In order to support this, the plateau concept was introduced.

Relationships can be established between the enterprise architecture models created at different moments in time and the migration models. Subsequently, analysis tools can be used to emphasize the differences between the different versions of models trough the linked plateaus. These differences are captured by the concept of gap. A gap is an important outcome of a gap analysis in Phases B, C, and D of the TOGAF ADM, and forms an important input for the subsequent implementation and migration planning. The gap concept is linked to two plateaus (e.g., baseline and target architecture, or two subsequent transition architectures), and represents the differences between these plateaus (Figure 15),

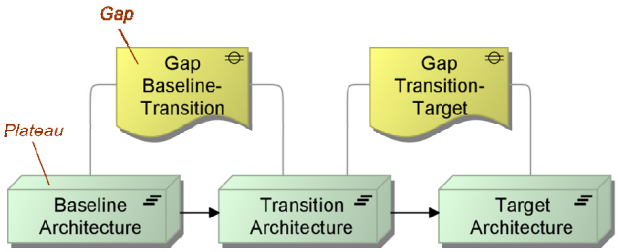

Figure 15. Migration Concepts

## VI. CONCLUSIONS AND FUTURE DIRECTIONS

TOGAF is a leading enterprise architecture method of The Open Group. ArchiMate has recently been adopted as an Open Group standard for modelling enterprise architectures. TOGAF and ArchiMate share their view on the use of viewpoints, and the concept of an underlying common repository of architectural artefacts and models; i.e., they have a firm common foundation. However, they complement each other with respect to the definition of an architecture development process and the definition of an enterprise architecture modelling language. ArchiMate provides a concrete visualization for the architectures and views proposed in TOGAF.

From the previous sections, it is clear that TOGAF and ArchiMate can be used in conjunction and cover much of the same ground. TOGAF itself provides no guidance on creating a consistent overall model of the architecture, but refers to tools that should provide this support ([8], Chapter 31):

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

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

This is where ArchiMate nicely complements TOGAF: it provides a vendor-independent set of concepts that would help to create a consistent, integrated model "below the waterline", which can be depicted in the form of TOGAF's views.

Presently, The Open Group is actively pursuing a closer integration between ArchiMate and TOGAF. An outline of this convergence is given by [4]. Some parts of TOGAF are not yet covered by ArchiMate concepts, and extensions to the language have been defined (and included in a proposal for version 2 of the standard) to fill these gaps. In particular, these concern on the one hand concepts for modelling the goals, motivations, principles and requirements used as inputs in defining an architecture, and on the other hand concepts for TOGAF's implementation and migration phases. With these two extensions, a next version of ArchiMate will have full coverage of TOGAF. Thus, these two complementary open standards will reinforce each other and help to advance the enterprise architecture discipline in general.

Future research is concerned with potential additional extensions of the language in other directions. In the practical use of ArchiMate, a number of fields have been identified in which such future extension of the language may be advisable: e.g., concepts for modelling business policies, decisions and rules, concepts for better support of the design process, or concpets that provide the link to (business) models at a more strategic level.

## REFERENCES


[1] W. Engelsman, H. Jonkers and D.A.C. Quartel, *ArchiMate® Extension for Modeling and Managing Motivation, Principles, and Requirements in TOGAF™*, White Paper, The Open Group, Feb. 2011.

[2] H. Jonkers, H. van den Berg, M.-E. Iacob & D. Quartel, *ArchiMate® Extension for Modeling the TOGAF™ Implementation and Migration Phases*, White Paper, The Open Group, Dec. 2010.

[3] J. van 't Woud, M. Waage, H. Hartman, M. Stahlecker and A. Hofman, *The Integrated Architecture Framework Explained*, Springer, 2010.

[4] H. Jonkers, E. Proper, M. Turner, *TOGAF and ArchiMate: A Future Together. A Vision for Convergence & Co-Existence*. Whitepaper, The Open Group, Nov. 2009.

[5] M.M. Lankhorst *et al*., *Enterprise Architecture at Work – Modelling, Communication and Analysis*, Second Edition, Springer, 2009.

[6] D.A.C. Quartel, W. Engelsman, H. Jonkers, 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.

[7] The Open Group, *ArchiMate® 1.0 Specification*, Van Haren Publishing, 2009. Also available on http://www.archimate.org

[8] The Open Group, *TOGAF™ Version 9*, Van Haren Publishing, 2009. Also available on http://www.togaf.org

[9] Object Management Group, Unified Modeling Language: Superstructure v2.0. OMG Document Number formal/05-07-04, August 2005

[10] Object Management Group, Business Process Modeling Notation, v1.1. OMG Document Number formal/2008-01-17, January 2008.

[11] M.M. Lankhorst, H.A. Proper, and H. Jonkers. The anatomy of the archimate language. *International Journal of Information System Modeling and Design (IJISMD)*, 1(1):1-32, 2010. 10.4018/jismd.2010092301

[12] P.S. Seligmann, G.M. Wijers, and H.G. Sol. Analyzing the Structure of I.S. Methodologies, an alternative approach. 1989.

[13] Business Motivation Model (BMM) Specification. Technical Report dtc/06-08-03, Object Management Group, Needham, Massachusetts, August 2006.

[14] D. Greefhorst and H.A. Proper. Architecture Principles - The Cornerstones of Enterprise Ar-chitecture. Enterprise Engineering Series. Springer, Berlin, Germany, 2011. ISBN-13 9783642202780

[15] H.A. Proper and D. Greefhorst. The Role of Principles in Enterprise Architecture. In H.A. Proper, M.M. Lankhorst, M. Schönherr, J. Barjis, and S.J. Overbeek, editors, Proceedings of the 5th Workshop on Trends in Enterprise Architecture Research, TEAR 2010, Delft, The Netherlands, volume 70 of Lecture Notes in Business Information Processing, pages 57-70. Springer, Berlin, Germany, November 2010.

[16] E.S.K. Yu and J. Mylopoulos. Understanding 'why' in software process modelling, analysis, and design. In Proceedings of the 16th international conference on Software engineering, Sorrento, Italy, Los Alamitos, California, pages 159-168, Los Alamitos, California, 1994. IEEE. ISBN-10: 081865855X

[17] G. Regev and A. Wegmann. Where do goals come from: the underlying principles of goal-oriented requirements engineering. In *In Proc. of the 13th IEEE International Conference on Requirements Engineering (RE05), Paris, France, August*, 2005.

[18] A. Van Lamsweerde. Goal-Oriented Requirements Engineering: A Guided Tour. In Proc. RE'01: 5th Intl. Symp. Req. Eng., 2001.

[19] Programme Management Based on MSP: A Management Guide, Jane Chittenden and Jan Van Bon. Van Haren Publishing, 2006. ISBN-13: 978-9077212677

[20] Project Management Body of Knowledge. Technical report, The Project Management Institute, November 2001.

[21] *Managing Successful Projects with PRINCE2*. The Stationery Office, 2009. ISBN-13: 9780113310593