



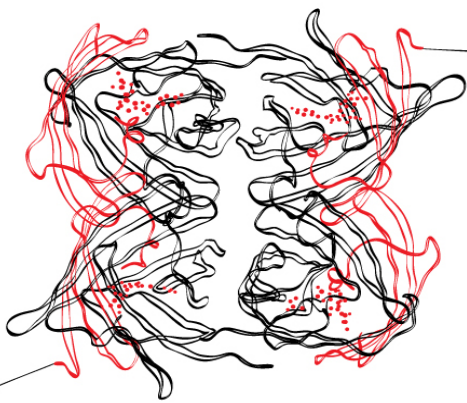
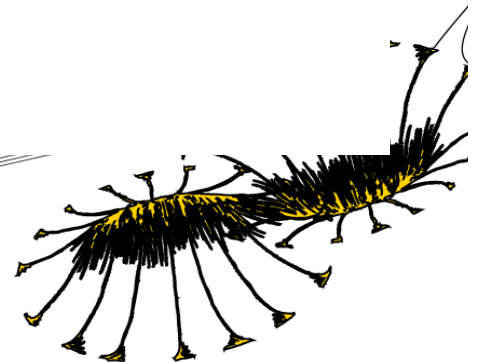
ENTERPRISE ARCHITECTURE TRANSFORMATION

ROADMAP PLAN ANALYSIS AND VISUALIZATION



Master Thesis
Business Information Technology

IWAN KURNIAWAN



UNIVERSITY OF TWENTE.

BiZZdesign



MASTER THESIS

ENTERPRISE ARCHITECTURE TRANSFORMATION ROADMAP PLAN ANALYSIS AND VISUALIZATION

Iwan Kurniawan

S1231626 | iwankurniawan@student.utwente.nl

Master of Science in Business Information Technology
School of Management and Governance
University of Twente
Enschede, the Netherlands
22 August 2014

Graduation Committee:

Dr. Maria-Eugenia Iacob, University of Twente
Dr. Ir. Marten J. van Sinderen, University of Twente
Dr. Ir. Dick A.C. Quartel, BiZZdesign

Acknowledgements

This thesis document is the final result of my master study and has been submitted in fulfillment of the requirement for the degree of Master of Science in Business Information Technology, University of Twente. The research has been conducted as a graduate internship assignment at BiZZdesign B.V., due to its commitment to help organizations worldwide to get a firm grip on change in an increasingly complex business reality, especially in the area of enterprise architecture. It has been a challenging and learning experience and I am deeply grateful to be given the opportunity to further understand the value of enterprise architecture to the organizations.

I would like to take this opportunity to express my gratitude to the Ministry of Communication and Informatics (Kemkominfo) Republic of Indonesia, for the scholarship granted. This two-year study has been made possible with the sponsorship, guidance and commitment provided and for that, I am greatly honored.

This master thesis, and the graduation internship, has been a wonderful experience as I was given the opportunity to learn from and work with the experts in the area of Enterprise Architecture. Maria Iacob and Marten van Sinderen, my two supervisors from the University, have provided insightful feedbacks, assistance and guidance throughout the thesis period. I would like to thank them for their significant inputs and suggestions in keeping me on track and forward. Likewise, I want to express my gratitude to Dick Quartel, my supervisor from BiZZdesign, for his direction and valuable advice in conducting this study. His continuous assistance and patience help me acquire clearer and better vision of the research direction, even when I was in doubts.

Finally, I would like to thank my friends and family for their endless supports and prayers. Also to the big family of PPI Enschede and IMEA for making Enschede feel warm like home. My graduate colleagues at BiZZdesign, thank you for the discussions and knowledge sharing sessions we had. It is always great to share the ups and downs while working on our research projects.

Iwan Kurniawan
Enschede, August 2014

Executive Summary

As the environment of the market is constantly changing, organizations are required to adapt to the technology advancement, demanding customers, aggressive competitors and regulatory changes. Aligning business and information technology is then an important factor to have this adaptation or transformation phase as effective as possible. Enterprise Architecture (EA) is a way to design and communicate the desired organizational changes related to the business strategy and to implement these changes across the operational structures, processes and systems of the organization's business and IT domain.

Study on depicting enterprises' architecture, including IT landscapes, is extensive. Although several researches have extensively elaborated on the enterprise architecture (e.g. comprehensiveness of the EA and maturity of EA development processes), the transformation phase has received little attention, especially on migrating from baseline architecture to target architecture. Therefore, this thesis aims to improve further on the enterprise architecture transformation, by focusing on the development of roadmap plan. The roadmap plan is intended to help visualize the alternative/possible paths of going from baseline architecture to target architecture.

The thesis identifies and analyses problems of the current guideline and support provided by EA framework, EA modeling language and selected existing EA tools. TOGAF and ArchiMate have been selected as the EA framework and modeling language, respectively, for their wide implementation in the industry and access openness to public. BiZZdesign Architect is used as the main EA tool of reference due to its certified alignment to the standards (TOGAF and ArchiMate). Two problems have been selected to be further addressed: (1) aggregating to a relation problem and (2) consolidated gaps, solutions and interdependency matrix problem.

The first problem derives from the fact that the existing ArchiMate definition of the plateau concept does not allow a relationship between two components to be aggregated (included) to the plateau. In practice, this condition is needed to show that the interactions among components, that are valid to a certain plateau, could also be valid and need explicit aggregation representation. The second problem is about the lack of clarity in dealing with the Consolidated Gaps, Solutions, and Dependencies matrix as described in the EA framework. The matrix is used as a planning tool when creating the work package. Although the actual matrix has been provided, the approach of how to prioritize the work packages could be made more concrete.

The thesis proposes solutions, or so called artifacts, to address these two selected problems. The first artifact is the extension or modification to plateau concept definition in ArchiMate. It accommodates the necessity of having aggregation relationship between the plateau and the relationship among the components belonging to the plateau. The second artifact is the gaps portfolio valuation in the form of 7-step approach. The approach is proposed to help the process of making the consolidated gaps, solutions and interdependency matrix more concrete. The end

result of the approach is the prioritized groups of the gap components which need to be closed by the enterprise in order to reach its target architecture.

The proposed approach is applied by means of a case study demonstration. ArchiSurance case study is used for the demonstration of the second artifact: gaps portfolio valuation. As for the evaluation purpose, three experts are consulted for their knowledge and opinion about the proposed solutions. Among the evaluation criteria used are correctness, completeness, feasibility, ease of understanding, and usefulness. In general, the experts agree that the solutions are needed in practice and provide sufficient level of correctness and completeness. Several remarks are given to further improve the solutions, such as treating relationship as a concept to avoid the complexity and confusion of ternary relationship. Finally, the thesis provides both academic and industrial contributions by proposing solutions which deal with both conceptual and practical concerns.

Table of Contents

Acknowledgements.....	i
Executive Summary	ii
Table of Contents	iv
List of Figures.....	vii
List of Tables	x
1 Introduction	1
1.1 Problem Statement	1
1.2 Research Goal.....	3
1.3 Research Questions	3
1.4 Research Methodology.....	4
1.5 Thesis Structure	7
2 Theoretical Framework.....	8
2.1 Key Concepts Definitions.....	8
2.2 Key User of Enterprise Architecture Transformation	9
2.3 Guidelines of Enterprise Architecture Transformation.....	13
2.3.1 TOGAF on EA Transformation	14
2.3.2 Archimate - Implementation & Migration Extension	20
2.4 Portfolio Valuation	25
2.5 Summary	30
3 Current EA Supports and Limitations.....	31
3.1 Roadmaps Visualization Support by EA Tools	31
3.1.1 IBM - Rational System Architect.....	33
3.1.2 Avolution – ABACUS.....	35
3.1.3 BiZZdesign – Architect	37

3.2 Limitations in Roadmap Support of EA Tool	39
3.3 Related Initiatives to EA Transformation Roadmap Plan.....	45
3.3.1 Visual Roadmaps for Managed Enterprise Architecture Evolution.....	45
3.3.2 Interactive Roadmap Generation.....	47
3.3.3 Modeling the Transformation of Application Landscape	48
3.4 Summary	50
4 Addressing the Selected Problems/Limitations	51
4.1 Selected Problems/Limitations.....	51
4.1.1 “Aggregating a relation” Problem.....	54
4.1.2 “Consolidated Gaps, Solutions and Dependencies Matrix” Problem.....	56
4.2 Solution to “Aggregating a Relation” Problem	58
4.2.1 Conceptual Solution	59
4.2.2 Practical Solutions.....	73
4.3 Solution to “Consolidate Gaps, Solutions and Dependencies Matrix” Problem.....	81
4.3.1 Gaps Portfolio Valuation.....	82
4.4 Summary	91
5 Demonstration	93
5.1 Case Study Method	93
5.2 ArchiSurance Case Study.....	93
5.2.1 Case Description.....	94
5.2.2 ArchiSurance Transformation Overview	95
5.2.3 Solution Implementation.....	100
5.3 Summary	112
6 Evaluation	113
6.1 Evaluation Dimensions	113
6.2 Interview	114
6.2.1 Interview Setting.....	115
6.2.2 Interview Question Script	116
6.3 Analysis and Result	117

6.3.1 Adding Relation Aggregation to Plateau Concept.....	118
6.3.2 Gaps Portfolio Valuation.....	120
6.4 Summary	123
7 Conclusion	124
7.1 Reviewing the Research Questions	124
7.2 Research Contributions	130
7.2.1 Theoretical Contributions	130
7.2.2 Practical Contributions	130
7.3 Research Limitations	131
7.4 Recommendations for Future Research	132
References	134
APPENDICES	138
Appendix A: The TOGAF Architecture Skills Framework	138
Appendix B: Architecture Development Methods (ADM) Overview	142
Appendix C: ArchiMate - Implementation and Migration Extension.....	146
Appendix D: Interview Transcripts	150

List of Figures

Figure 1: DSRM Possible Entry Points (Peffer et al., 2007)	5
Figure 2: Thesis Outline	7
Figure 3: Architecture Development Methods of TOGAF (The Open Group, 2011).....	14
Figure 4: Components of ArchiMate Approach (Iacob et al., 2012)	21
Figure 5: Implementation and Migration Extension Model (The Open Group, 2012).....	22
Figure 6: Work Package (The Open Group, 2012)	22
Figure 7: Plateau (The Open Group, 2012)	23
Figure 8: Bedell and Enterprise Architecture (Quartel et al., 2010).....	27
Figure 9: Calculating Effectiveness and Importance (Buschle & Quartel, 2011)	28
Figure 10: Gartner Magic Quadrants for EA Tools (Gartner, 2013)	32
Figure 11: Additional Concepts to Migration and Implementation Extension (Owen, 2013)	34
Figure 12: Milestone View Showing Work Package with Status (Owen, 2013)	34
Figure 13: Additional Lifecycle States to Concepts (Owen, 2013)	35
Figure 14: Example of Application Catalogue (ABACUS, 2013)	36
Figure 15: Application Roadmap Dashboard (ABACUS, 2013)	37
Figure 16: Roadmapping Browser of BiZZdesign Architect	38
Figure 17: Plateau Assignment through Properties	38
Figure 18: Plateau Assignment through Aggregation	38
Figure 19: Gap Analysis of Application Architecture, ArchiSurance Case.....	40
Figure 20: Gap Analysis of Technology Architecture, ArchiSurance Case.....	40
Figure 21: Project Context Diagram, ArchiSurance Case.....	41
Figure 22: Possible Transformation Path, ArchiSurance Case	42
Figure 23: Sequential Roadmap Plan, ArchiSurance Case	42
Figure 24: Aggregating a Relation Problem.....	43
Figure 25: Updating Component Problem	43
Figure 26: Date Validity Checking Limitation	44
Figure 27: Business Support Migration Roadmap Plan (Buckl et al., 2009).....	46
Figure 28: Information Model (Buckl et al., 2009)	46
Figure 29: Planning Components in Context (Diefenthaler, 2013).....	48
Figure 30: Relationships between Implementation & Migration Extension and the ArchiMate Core Concepts (The Open Group, 2012)	55
Figure 31: Aggregation and Nesting Way of Modeling (The Open Group, 2012)	56

Figure 32: Generic Metamodel: The Core Concepts of ArchiMate (The Open Group, 2012).....	60
Figure 33: Extension of Relationships between Implementation & Migration Extension and the ArchiMate Core Concepts	62
Figure 34: Composition relationship in aggregation extension.....	62
Figure 35: Junction relationship in aggregation extension	63
Figure 36: Specialization relationship in aggregation extension	63
Figure 37: Example of aggregating a relation between components	64
Figure 38: Scenario 1 – Target and source components	64
Figure 39: Scenario 2 – Target and source components	65
Figure 40: Scenario 3 – Target and source components	65
Figure 41: Scenario 4 – Target and source components	65
Figure 42: Grouping relationship for plateau definition	67
Figure 43: Plateau - grouping relationship	67
Figure 44: Architectural element's properties and relationship attributes	69
Figure 45 : Plateau aggregation relationship: specialization and notation.....	69
Figure 46: Proposed type of aggregation relationship	70
Figure 47: Call center application plateau aggregation relationship.....	70
Figure 48: Illustration of plateau duplication functionality.....	74
Figure 49: Plateau definition through profiling	75
Figure 50: Plateau definition through aggregation relationship	76
Figure 51: Two ways plateau definition in BiZZdesign Architect	76
Figure 52: Multiple components aggregation to plateau	77
Figure 53: Total view to define plateau aggregation	77
Figure 54: Plateau concept and its definition view	78
Figure 55: Gap, Plateau and Core element interrelation, derived from The Open Group (2012).....	79
Figure 56: Gap concept classification.....	80
Figure 57: Illustration of gap analysis of plateaus transformation	80
Figure 58: Illustration of gap concept classifications.....	81
Figure 59: Example of gaps analysis view.....	83
Figure 60: Goals – components relationship	84
Figure 61: Filtered model for Gaps Components Group	86
Figure 62: Determining strategic importance scores (based on Bedell, 1985)	87
Figure 63: Example of interdependency level of gap components.....	89
Figure 64: ArchiSurance Organization Structure (Jonkers et al., 2012a)	94

Figure 65: Overview of ArchiSurance's as-is architecture.....	96
Figure 66: Overview of ArchiSurance's future architecture.....	97
Figure 67: General overview of gaps.....	98
Figure 68: Affected architectural components in the transformation process.....	98
Figure 69: Overview of application architecture differences of ArchiSurance	99
Figure 70: Overview of technology architecture differences of ArchiSurance	99
Figure 71: Group 1 – gap components.....	101
Figure 72: Group 2 – gap components.....	102
Figure 73: Group 3 – gap components.....	102
Figure 74: Group 4 – gap components.....	102
Figure 75: Group 5 – gap components.....	102
Figure 76: Group 6 – gap components.....	103
Figure 77: Group 7 – gap components.....	103
Figure 78: Group 8 – gap components.....	103
Figure 79: Group 9 – gap components.....	103
Figure 80: Fragment of goal, principles and requirements of organization.....	104
Figure 81: Filtered-out model of gaps components & goals.....	105
Figure 82: Allocated level of importance and effectiveness of gaps components	105
Figure 83: Interrelations of the architectural components	107
Figure 84: Gaps' groups mapping	111
Figure 85: Alternative transformation paths.....	112
Figure 86: IS Success Model (DeLone & McLean, 2003).....	113
Figure 87: Implementation and Migration Extension Metamodel	146
Figure 88: Relationship between Implementation and Migration Extension and the ArchiMate Core Concepts	146
Figure 89: Relationship between Plateau, Deliverable and Motivation Concepts.....	146
Figure 90: Concepts and Relationships of Project Viewpoints.....	148
Figure 91: Concepts and Relationships of Migration Viewpoints	148
Figure 92: Concepts and Relationships of Implementation & Migration Viewpoints.....	149

List of Tables

Table 1: Design Science Research Methodology (Peppers et al., 2007)	4
Table 2: Implementation and Migration extension Concept (The Open Group, 2012).....	23
Table 3: Result of Modeling Approaches Evaluation (Hofer, 2013).....	50
Table 4: ArchiMate Relationships (The Open Group, 2012)	61
Table 5: Summary of alternative solutions assessment.....	73
Table 6: Proposed steps of gaps portfolio valuation	82
Table 7: Viewpoints on architecture performance (Iacob & Jonkers, 2009)	88
Table 8: Gaps components assessment	90
Table 9: Group gaps components assessment	91
Table 10: Collected gaps components	101
Table 11: Gaps components – Goals Mapping.....	104
Table 12: Gaps components level of importance.....	106
Table 13: Gaps components level of effectiveness	107
Table 14: Gaps components' level of interdependency	108
Table 15: Overall scores of gaps groups	109
Table 16: Interview setting	115
Table 17: Interview questions – Evaluation sessions.....	116
Table 18 : Summary of evaluation session.....	117
Table 19: Proficiency Level.....	138
Table 20: Generic Skills	138
Table 21: Business Skills and Methods.....	139
Table 22: Enterprise Architecture Skills.....	139
Table 23: Program or Project Management Skills	140
Table 24: IT General Management Skills	140
Table 25: Technical IT Skills	141
Table 26: Legal Environment	141
Table 27: ADM Phases and Steps	142
Table 28: Description of Project Viewpoint.....	147
Table 29: Description of Migration Viewpoint	148
Table 30: Description of Implementation and Migration Viewpoint.....	149

1 Introduction

This chapter aims to provide background information regarding the research area. Section 1.1 discusses the problem statement related to the enterprise architecture transformation roadmap. Section 1.2 presents the research goal of the thesis and in order to meet the goal; section 1.3 formulates the research questions. Section 1.4 describes the research methodology used to address the research questions. Finally, section 1.5 outlines the structure of the thesis and briefly explains the purpose of each chapter.

1.1 Problem Statement

Many organizations, whether they operate in public or private sectors, have to deal with the constantly changing environment driven by various factors, such as technology advancement, demanding customers, aggressive competitors as well as regulatory changes. In order to remain competitive, these organizations need to adapt by swiftly changing their business strategy and/or their business process. To do so, organizations need to have the overall overview of and the impact on the organization. Furthermore, communication means to steer effectively the change process, the involvement of the people and the optimum coordination of resources are necessary (Iacob et al., 2012).

Enterprise Architecture (EA) can be used as a means to design and communicate the desired organizational changes according to the business strategy, and to implement these changes across the operational structures, processes and systems of the organization's business and IT domains (Ross et al., 2006). EA is defined as the complete, consistent and coherent set of methods, rules, models, and tools that will guide the (re)design, migration, implementation and governance of business processes, organizational structures, information systems and the technical infrastructure of an organization based on a vision (Iacob et al., 2007). When organizations have clear picture of their enterprise architecture, they will manage their assets better and plan the necessary changes according to their strategy more easily.

Architecture roadmaps are used to describe the path (or journey) of change, over a certain period of time, from the current situation (baseline architecture) to the desired situation (target architecture). This could be used as a guideline in monitoring the change process (enterprise architecture transformation) by analyzing the gap between the target and baseline architectures. Buckl et al (2009) state that many of today's enterprises face problems in managing the transformation from a current EA to an envisioned EA via intermediary planned architectures. Aier et al. (2009) identify several causes with regards to this: missing practical methodologies for architecture roadmapping, inadequate representation of the concept of time in architectural models and insufficient tool support for architecture planning.

A basis to develop enterprise architecture in a consistent and standardized manner has been provided by the enterprise architecture framework. This is with regards to ensure that the various descriptions of architectures developed within the enterprise, and most probably by

different architects, support the comparison and integration of architectures across multiple domains (business, data, application and technology). However, the fact that organizations within an enterprise might possess different levels of architecture maturity and technology capability results in difficulties for a single enterprise architecture tool to satisfy all organizations' needs. Therefore, in order to make the enterprise architecture management successful, the architects, most of the time, harmonize their architecture tools with their architecture maturity level, team capability and focus. In addition, it is very challenging for a single tool to accommodate a variety of architecture development maturity levels and specific needs across an enterprise.

In addition, an extensive analysis of EA management tools has been performed in the form of tools survey by Sebis (Buckl et al., 2008). The survey was conducted in cooperation with 30 industry partners and analyzed the EA tools produced by nine major players in the market. The survey pursued a threefold evaluation approach. The first set of scenarios focused on the functionalities that should be provided by EA tools, such as creating visualization, information model flexibility and usability. The second set of scenarios evaluated the essential constituents of EA management like project portfolio management, IT architecture management and business object management. These two set of scenarios were complemented by an online questionnaire.

The study concluded that, based on the evaluation results, EA tools lack in the capability of supporting the automated creation of the visualizations. It should be noted that some tools provided support in generating the future architecture visualization which illustrate the architecture at a given time in the future (snapshot view). The snapshot view is related to the static complexity of the constituents and dependencies which were handled well by the EA tools vendor through visualization and collaborative maintenance functionalities. The dynamic aspect of EA planning resulted from the changes over time, however, was not addressed well by the EA tools. As a result, roadmapping, versioning and transformation paths are insufficiently supported.

Most of the EA tools support the transformation process by displaying the gap view as a result of gap analysis between two states of architectures. However, the gap is mostly focusing on the components that belong or do not belong to certain state of architecture. It does not yet cover the relations between components within the architecture state. For example, two architecture states might consist of, among others, two exactly the same components. But the relation between the two components might be removed in the new state of the architecture. And thus, this relation removal is not captured in the gap analysis.

Although general guidelines and insights of performing the transformation process have been provided by many frameworks, such as TOGAF, the implementation support by EA tools is still limited. Moreover, analyzing the relationships between components in EA state needs further support development. Therefore, improving the development of a roadmap plan by EA tools is needed.

1.2 Research Goal

The main goal of the research is to further improve the development process of a roadmap plan for enterprise architecture transformation. The roadmap plan is intended to help visualize the alternative/possible paths of going from baseline architecture to target architecture. In other words, it aims to model the process of visualizing and analyzing the roadmap of the EA transformation. By carefully analyzing all relationships between components within the states of architecture, the roadmap plan tries to support the users in their transformation planning process.

1.3 Research Questions

In meeting the above research goal, the main research question which is divided into sub-research questions, is formulated as follow:

Main Research Question:

How can EA transformation be improved in the form of a roadmap plan?

To guide the study and the research, the following sub-research questions need to be addressed and are answered throughout the research. In order to propose an improvement to the EA transformation roadmap plan, the current state-of the art or the existing guideline provided by EA frameworks need to be studied. The guidelines would be specifically observed in the area of transformation process.

Sub-research Questions:

- **RQ1:** What do the Enterprise Architecture frameworks say about the transformation process?
 - Who is the key user of roadmap plan and what is the main function of roadmap plan according to this key user?
 - What are the step-by-step guidelines in developing the roadmap plan?

Identifying the key user of the roadmap plan is essential in shaping the direction of the proposed solution. This is imperative so that the outcome of the research would provide meaningful applicability in real case implementation. Furthermore, exploring the current step-by-step guideline provided by the EA frameworks in developing the roadmap plan provides the basic foundation or information on the general guideline. This also allows the research to identify some problems already encountered while exploring the existing guidelines and determining whether the guideline is sufficient.

- **RQ2:** How is the Enterprise Architecture transformation currently supported by the Enterprise Architecture tools?
 - What are the limitations of the Enterprise Architecture tools in analyzing and visualizing the Enterprise Architecture transformation?

After finding out how the transformation process should be performed, exploration on the current support of EA tools is the next consideration. Translating the guidelines into real implementation of transformation process is supported by the EA tools. To do this explorative study, a selection of the existing EA tools is needed in order to maintain the focus and deal with time limitation. For that purpose, only several leading EA tools available in the market are selected. The research is interested in identifying what the limitations are in terms of analysis and visualization perspective towards the roadmap plan development.

- **RQ3:** How could the development of roadmap plan be improved in ArchiMate and Architect?

The problems that are identified by the research question 2 above need to be grouped and categorized. The next step is to select which of the problems would be addressed extensively by the research. Depending on the problem selection, a proposed solution to improve the roadmap plan development in ArchiMate and Architect would be designed. .

1.4 Research Methodology

The work of Peffers et al.(2007), Design Science Research Methodology (DSRM), will be used in this research in which the six activities or steps are followed sequentially. The table 1 below depicts the steps with the description about the activities performed during each step and the knowledge base on how to execute the steps. The objectives of DSRM are to provide a nominal sequential process to conduct design science research, to build upon prior literature about design science in information science and reference disciplines and to provide research with a mental model or template for a structure for research outputs. Therefore, it aims to provide an easy to understand structure to conduct a design science research.

Table 1: Design Science Research Methodology (Peffers et al., 2007)

DSRM Activities	Activities Description	Knowledge Base
Problem identification and motivation	<i>What is the problem?</i> Define the research problem and justify the value of a solution	Understand the problem relevance and its current solutions and their weaknesses.
Define the objectives of a solution	<i>How should the problem be solved?</i> In addition to general objectives such as feasibility and performance, what are the criteria that a solution for the problem defined in step one should meet?	Knowledge of what is possible and what is feasible. Knowledge of methods, technologies, and theories that can help with defining the objectives.
Design and development	<i>Create the artifact that solves the problem.</i> Create constructs, model, methods, or instantiations in which a research contribution is embedded.	Application of methods, technologies, and theories to create an artifact that solves the problem.

Demonstration	<i>Demonstrate the use of the artifact.</i> Prove that the artifact works by solving one or more instances of the problem.	Knowledge of how to use the artifact to solve the problem.
Evaluation	<i>How well does the artifact work?</i> Observe and measure how well the artifact supports a solution to the problem by comparing the objectives with observed results.	Knowledge of relevant metrics and evaluation techniques.
Communication	<i>Communicate</i> the problem, its solution, and the utility, novelty and effectiveness of the solution to researchers and other relevant audiences.	Knowledge of disciplinary culture.

Case study would be demonstrated in measuring the efficacy of the artifact to solve the problem. At the end of evaluation step, the research might decide whether to iterate back to step 3 or try to improve the effectiveness of the artifact or to continue on to communication and leave further improvement to subsequent projects. The communication step in this research will be done in the form of colloquium presentation for the graduation and the publication of the final research documentation by the University of Twente.

Although the research process is presented in sequential activities, the researcher is not mandated to follow the exact order from step 1 through step 7. In practice, a particular research might start at almost any step and move outward. There are four possible entry points of doing design science research according to Peffers et al. (2007): problem-centered approach, objective-centered solution, design and development centered approach and observing a practical solution that worked. These entry points are depicted in Figure 1 below.

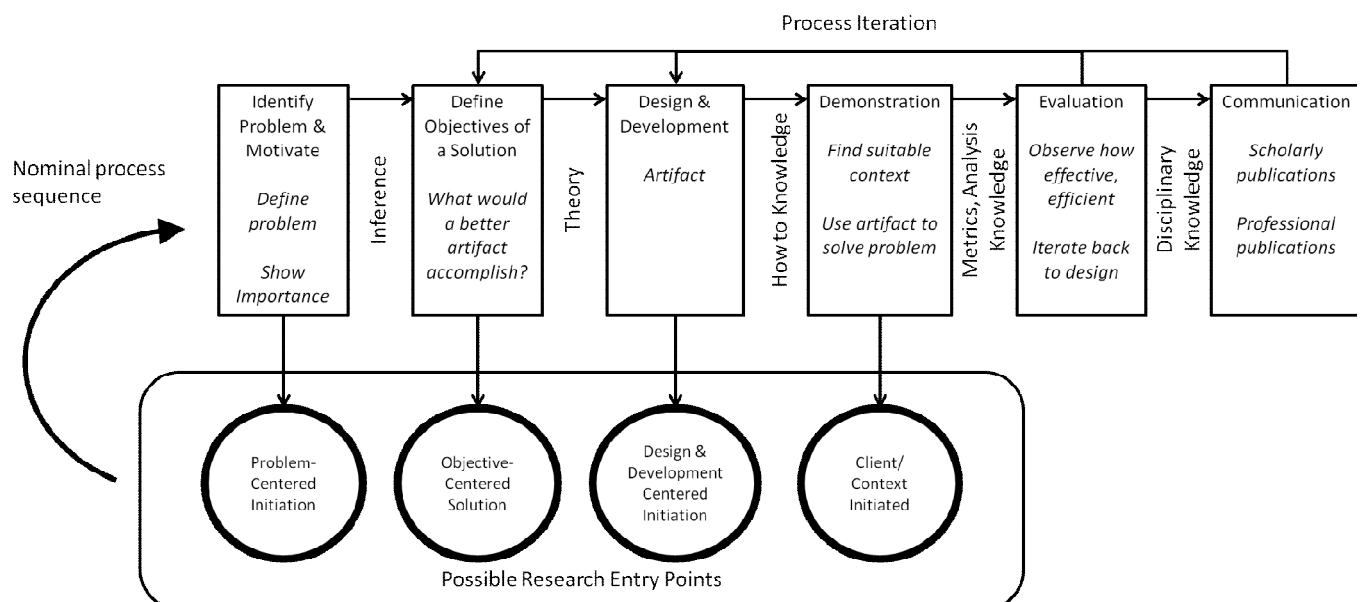


Figure 1: DSRM Possible Entry Points (Peffers et al., 2007)

A problem-centered approach is the normal or standard research process where it starts from step 1. The research could be the result of the observation of the problem or it is derived from the suggestion of a prior research project. An object-centered approach, which starts at step 2, could be driven by the product of consulting experience where the result of a project does not meet the client's expectation and a better job performance is wished for. A design and development centered approach, starting at step 3, would be the result of a situation where the existing artifact is not fully completed or has not been formally thought through. Such artifact might come from another similar research domain or have been used to solved different problem, or appear as an analogical idea. Lastly, observing a practical that worked starts at step 4 and is the result in a design science solution if researchers work backwards to apply rigor to the process retroactively.

Referring back to the background of the research as stated in previous sub-section, the approach taken by this research is perceived as a problem-centered approach. And thus, the research follows a nominal sequential process starting at step 1. The research questions outlined previously are addressed by different research approaches:

1. Exploratory literature reviews

In order to have firm understanding of the research, in-depth literature studies were conducted. Based on this, addressing established and relevant sources in the field of EA transformation, the main concepts used in this research are described. First, the overall guideline of EA transformation process is elaborated. Subsequently, current support to roadmap plan by EA tools is discussed, including the limitations identified. This leads to the proposed improvement of roadmap plan development in guiding the EA transformation.

2. Interview

The interviews were conducted with internal EA consultant(s) to gain more understanding of the current support provided by the EA tool with regards to transformation process. Lessons learned and case examples from previous and existing clients are discussed to identify and propose potential improvement to the tool support. Based on the interviews, as well as the studied literature, the proposed solution is designed to improve the tool. Interviews were also conducted to validate the proposed solution and to evaluate the usability of the solution in practical scenarios.

3. Case studies

To understand a complex issue and to add strength of the experience from previous research, case studies can be considered as a good technique which emphasizes detailed contextual analyses of a limited condition and their relationship. As Yin (1993) defines that case study research method as an empirical inquiry that investigates a phenomenon within its real-life context using multiple sources of evidence, and the boundaries between phenomenon and context are not clearly evident. Case studies were also used to show the applicability of the proposed solution. Hence, the demonstration process of the solution was also conducted partly by the case studies.

Since the thesis research was conducted at BiZZdesign, the scope of the research and the proposed solution is mostly based on the EA framework and EA tool applied by BiZZdesign, which are TOGAF (The Open Group Architecture Framework), ArchiMate (the EA modeling language) and Architect (EA tool, developed by BiZZdesign).

1.5 Thesis Structure

The thesis consists of four main parts, each describing an important research phase depicted in several chapters as illustrated in Figure 2. The background phase is important to gain an understanding of the motivation as well as the structure of the research. Chapter 1 introduces the readers to the motivation and structure of the research where it includes the problem statement, research questions as well as the research structure. Chapter 2 presents the related main concepts of EA transformation discussed and investigated in this research. The chapter discusses the key user of the subject of research. Existing guidelines provided by EA frameworks on how to perform the EA transformation process are elaborated.

Chapter 3 identifies and analyses problems or limitations of the current support provided by the selected three existing EA tools. Some limitations and potential improvements of EA tools in its roadmap plan support are identified. The chapter also summarizes several current approaches of related work in translating the EA transformation guideline into practical roadmap plan. The solution phase aims to design a proposed solution in roadmap plan development in analyzing and visualizing EA transformation. The solution phase is mainly elaborated in Chapter 4, based on the concepts and the identified and selected limitations.

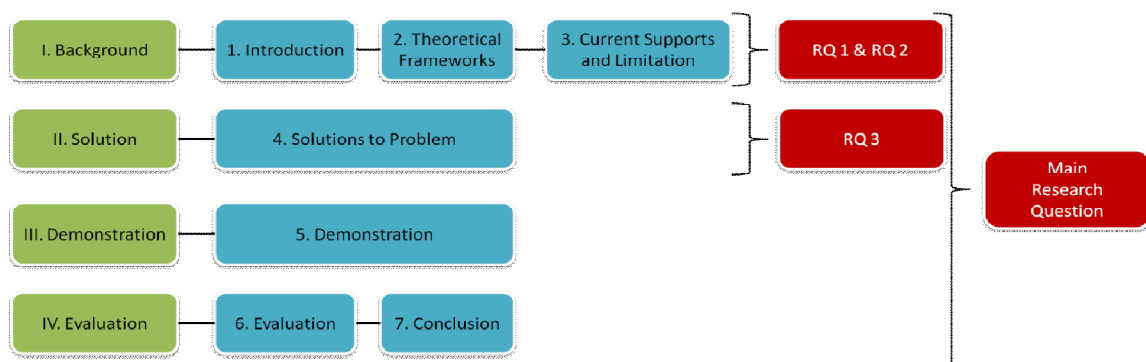


Figure 2: Thesis Outline

The demonstration phase, as presented in Chapter 5, aims to show the operationalization of the proposed solution in the organization-specific context by performing case study. Evaluation phase to the artifact simulation is described in Chapter 6. Further interview with experts is conducted to evaluate the usability of the solution in supporting the key users. The proposed improvement solution is observed to be able to know how well it supports a solution to the problem. In chapter 7, a general conclusion is drawn, answering the research questions. Furthermore, the limitations of this research and recommendations for future research are presented.

2 Theoretical Framework

This chapter provides literature review to understand the key concepts and the current situation of the research topic. This chapter addresses research question 1 about what the Enterprise Architecture frameworks say regarding the EA transformation process. Section 2.1 provides general understanding and definitions of key concepts with regards to enterprise architecture transformation. Section 2.2 discusses the key user of the enterprise architecture transformation, including its roles and expectations in dealing with the transformation. Section 2.3 explains how the transformation processes are defined by the EA frameworks. Key concepts of view and viewpoint with regards to transformation process introduced by the ArchiMate, EA modeling language, will be discussed. Section 2.4 describes about the IT projects portfolio valuation in EA setting. This valuation area provides insights on how to measure the value of architectural elements so that they could be assessed and analyzed for further decision making process. Finally, section 2.5 presents the general summary of this chapter.

2.1 Key Concepts Definitions

As previously stated, EA is a means to design and communicate the desired organizational changes according to the business strategy. EA goes further than only designing and communicating, it is also implementing these changes across the operational structures, process and systems of the organization's business and IT domains (Ross et al. 2006). To provide more firm basic understanding, definitions of several fundamental elements need to be in place.

Many definitions of architecture exist. Schekkerman (2008) defines architecture as the structure of components, their (inter)relationship, and the principles and guidelines that govern the design and evolution over time. Similar definition is given by Hilliard (2000) where architecture is defined as the basic organization of a system that is embodied in its components, relationship among them and to the environment, as well as the guiding principles on the design and evolution. Back in 1996, Zachman described architecture as the set of descriptive representations that are relevant for describing an enterprise and that can be produced to management's quality requirements and that can be maintained over the period of its useful life (change). To summarize, **architecture** contains components or elements, relationships among them, the environment they interact with, the principle/guideline and the representation of these components.

Generally, an **enterprise** could be defined as a collection of organizations with common objectives/goals and principles. An enterprise could be the whole corporation or only part of the corporation (a division of corporation). An enterprise could be a government organization or private/commercials organization or even a network of geographically separated organizations that are connected together by common objectives.

Likewise, there exist definitions of what **Enterprise Architecture** (EA) means. Lapkin (2007) defines EA as a process of translating business vision and strategy into effective enterprise

change by creating, communicating and improving the key principles and models that describe the enterprise's future state and enable its evolution. EA, as defined by Iacob et al. (2007), is the complete, consistent and coherent set of methods, rules, models, and tools that will guide the (re)design, migration, implementation and governance of business processes, organizational structures, information systems and the technical infrastructure of an organization based on a vision (Iacob et al, 2007). EA is perceived as a management tool to help translate the goal of an organization from the current (as-is situation) state to the future (to-be situation) state (Lankhorst, 2009). Some key aspects of EA are model-based approach, evolution of organization (enterprise change), and decision support for IT related issues. Similarity of these definitions of EA is that it guides the evolution or change process from one state to another.

Current state of the EA, or referred to Baseline Architecture in this research, is the initial state or the as-is condition of the EA. **Baseline architecture** can be defined as the set of products that portray the existing enterprise, the current business practices and the technical infrastructure. Based on the strategy and business goals, some developmental and incremental processes need to take place to reach the to-be state of the EA, or referred to Target Architecture, in this research. Thus, **target architecture** can be defined as the set of products that portray the future or end-state enterprise, generally captured in the organization's strategic thinking and plans. In the evolution journey, the change process is not a single quantum step but rather incremental. Transition Architecture is then defined as the state of the EA in between the change process before reaching the Target Architecture. An enterprise might experience more than one transition architectures during its evolution journey.

A **roadmap** is the abstracted plan for the business or technology change, typically operating across various disciplines and over multiple years (The Open Group, 2012). Architecture roadmaps are used to describe the path (or journey) of change, over a certain period of time, from the current situation (baseline architecture) to the desired situation (target architecture). This could be used as a guideline in monitoring the change process (enterprise architecture transformation) by analyzing the gap between the target and baseline architectures. Timeline view is then necessary in describing or visualizing the architecture roadmap in order to show the required activities needed to be performed to realize the target architecture.

Roadmap plan, according to Schekkerman (2008) is very important and is considered as a primary tool for program management and investment decision. This is because it holds the data about the current, under way and planned architectures which are making up the development programs of an organization.

2.2 Key User of Enterprise Architecture Transformation

Knowing who the key users of enterprise architecture in general and enterprise architecture transformation in specific is important to understand what their interests are with regards to the transformation process. By doing so, it would be more structured and focused in describing the process of enterprise architecture, including the analysis and visualization of the roadmap plan.

A key user could be defined as a stakeholder. According to Minoli (2008), a stakeholder could be an individual, a group of people or an organization which possesses key role in the architecture. Moreover, a stakeholder must have concern about or interest in the architecture (Iacob et al., 2012; Hilliard 2000), or is involved in the process of creating or using the architecture. Different stakeholders will have different interests which could be conflicting to one another. That is why; in most cases stakeholders are only concerned about the impact of the architecture to their specific interests (Iacob et al., 2012; Jonkers et al., 2006, 2012).

Hilliard (2000) categorizes two kinds of enterprise architecture stakeholders: the architects and the acquirer of the architecture. Foorthuis et al., (2010) also classify the enterprise architecture stakeholders into two main groups: the creator and the user of enterprise architecture. The creator of enterprise architecture could be enterprise architect business and information, enterprise architect application and infrastructure, manager and external enterprise architecture consultant. Whereas the user of enterprise architecture could be in the role of manager, project manager, project architect, business analyst/designer, system & information analyst/functional designer, software architect, technical designer, developer/programmer and maintenance engineer.

In practice, roadmap plan of enterprise architecture transformation is created by the enterprise architects after collecting the necessary inputs from various stakeholders or roles, such as business architect, application architect, technology or infrastructure architect, program or portfolio manager and many others. The difference between enterprise architect and other types of architects is that the enterprise architect covers a wide range of business and IT while domain architects focus on one aspect of the enterprise (business, application, data) and solution architects focus on one small part of the implementation of the architecture (applications, software, business processes).

In turn, the enterprise architect in communicating the roadmap plan of enterprise architecture transformation must consider the interested stakeholders' concerns in deciding what to display in a roadmap plan. This is to ensure that the information given in the roadmap plan is adjusted to the need.

The enterprise architect, as defined in TOGAF, is responsible to ensure the completeness of the architecture. It means the fitness-for-purpose of sufficiently addressing various concerns of the stakeholders. Ensuring the integrity of the architecture in terms of connecting multiple views to each other and satisfactorily handling the conflicting concerns among the stakeholders are also under the responsibility of the enterprise architect.

Even though enterprise architect has professional relationship with executives of the enterprise to gather and articulate the technical vision and to produce strategic plan for realizing it, enterprise architect does not create the technical vision of the architect. Documentation of design decisions for application development teams or product implementation team to execute must be produced by the enterprise architect. A key point to emphasize about enterprise

architect is that enterprise architect is not the builder and it must remain at abstract level to ensure that it does not get into practical implementation. The enterprise architect responsibility is to know and focus on the critical few details and interfaces that really matter and not to get trapped and overloaded with the rest.

In summary, the roles of enterprise architect are as follow:

- Understand and interpret requirements. The enterprise architect involves in the discovery and documentation of the customer's business scenarios that are driving the solution. It must understand the requirements and translate them into the architecture specification.
- Create a useful model. Well formulated model of the components of the solutions is developed based on the collected requirements. The model should be augmentable to fit various circumstances by having multiple views to communicate the ideas effectively. It is then under the responsibility of the enterprise architect to maintain the integrity of the model. It provides and maintains the models as a framework to guide what should be done within and/or outside the organization.
- Validate, refine and expand the model. In order to further improve and define the model, assumptions must be verified by involving subject matter experts. To make the result more flexible and linked to current and expected requirements, new ideas could be added.
- Manage the architecture. This role is fulfilled by continuously monitoring the models and updating them according to the changes, additions or alterations. During the development and decision points of the program, enterprise architect must represent the architecture and issues. By doing so, architect fosters the information sharing about customers, technical and architecture between organizations.

In performing its roles effectively, enterprise architect is expected to have some relevant competencies. Land et al., (2009) distinguish two kinds of essential competencies relevant for the enterprise architect: professional and personal competencies. Professional competencies are competencies that are dealing with knowledge, attitude and skills necessary to a successful performance in a specific function or role. Since enterprise architect covers breadth of business and IT as compared to domain architects and solution architects, enterprise architects are therefore required to have understanding in all four domains (business, information, information systems, and infrastructure). TOGAF lists several skills set in the architecture skills framework as follow:

- Business skills and methods, typically comprising business cases, business process, strategic planning. Extensive and substantial practical experience and applied knowledge on the subject must be possessed by the enterprise architect. This business skills and methods are considered important and critical for an enterprise architecture role.
- Enterprise architecture skills, typically comprising modeling, building block design, applications and role design, system integration. Like the business skills and methods skills, enterprise architecture skills are considered fundamental for the enterprise architect role to perform its tasks. Therefore, and extensive and applied knowledge is

required.

- Program or project management skills, typically comprising managing business change, project management methods and tools. Although the skills set is considered important, the enterprise architect does not have to acquire an extensive level of understanding. Moreover, coordination and communication with project or program managers is critical with regards to these competencies.
- IT general knowledge skills, typically comprising brokering applications, asset management, migration planning, service level agreement. These skills set require enterprise architect to possess good detailed knowledge of subject area and to be capable of providing professional advice and guidance. Enterprise architect must be able to integrate capability of other domain architects (applications, data, technology / infrastructure) into architecture design.
- Technical IT skills, typically comprising software engineering, security, data interchange, data management. Like the IT general knowledge skills, technical IT skills should be possessed at the similar level by the enterprise architect. The difference in these skills set is that the enterprise architect should pay more attention in coordinating and communicating with the technology / infrastructure architect.
- Legal environment, typically comprising data protection laws, contract law, fraud. Apart from the data protection law knowledge, enterprise architect is required to possess a normal level of awareness about issues related to this competency. Enterprise architect is required to understand the background, issues and implications to know how to proceed further or to advice accordingly.

Personal competencies are related to the competencies that can be used in several functions or roles, such as communication skills, and personality characteristics. This type of competencies is important because in delivering the tasks, enterprise architect must interact with different kinds of stakeholder including the management or business level stakeholders as well as the domain-specific stakeholders or technical level stakeholders. No distinction is made between different types of architect with regards to personal competencies.

Creativity and leadership are considered as crucial personal competencies because enterprise architect needs to cover the whole spectrum of business and IT and most of the time operates in a leadership role in collaborating with other architects. Moreover, enterprise architect is dealing with change management in performing its tasks related to EA transformation. Communicating and transferring the knowledge of roadmap plan to various stakeholders requires good level of analytical, communication and negotiation skills.

By considering the enterprise architect's professional and personal competencies, the roadmap plan of enterprise architecture transformation should be able to cover various levels of details. This is to support the fact that enterprise architect must have broad understanding about different domains. The roadmap should be able to link between the work packages and the goals and should be able to detail out the related architectural components impacted by the deliverables of the work packages. This emphasizes the importance of roadmap plan in not bringing or visualizing too many (technical) details on a single view but being able to further

detail out the necessary (technical) information when needed. This flexibility factor is important for an effective roadmap plan communication to various stakeholders.

2.3 Guidelines of Enterprise Architecture Transformation

An enterprise architecture framework provides description and documentation of the underlying infrastructure of an enterprise. And thus, it provides a basis for the hardware, software and network together. Such architecture documentation is imperative to ease the maintenance as well as improvement so that the system is not obsolete before it is even built.

There are numerous architectures and architectural frameworks in use today. Although those frameworks may overlap in certain addressed areas, they have been designed to address specific needs or concerns, according to the stakeholders and their concerns. Some frameworks have a very specific scope and therefore are only applicable to that specific context. Some might be truly enterprise oriented and some are specific only to the development of IT system. Urbaczewski (2006) compared some enterprise frameworks based on their views, abstractions, and coverage of the system development life cycle. Among the architectural frameworks being compared are Zachman, DoDAF (Department of Defense Architecture Framework), FEAF (Federal Enterprise Architecture Framework), TEAF (Treasury Enterprise Architecture Framework) and TOGAF (The Open Group Architectural Framework).

Published in 1987, the Zachman framework for enterprise architecture is considered as the pioneer in the area. It is based on the principles of classical architecture that establishes a common vocabulary and set of perspective for describing not-simple enterprise system. It has two dimensions; six perspectives of views (planner, owner, designer, builder, subcontractor and user) and six basic questions (what, how, where, who, when and why). It does not detail out on the sequence, process or implementation but rather focuses on ensuring that all views are well established. Nor does it have an explicit compliance rules since it is not a standard written by or for a professional organization (Urbaczewski, 2006). According to Lankhorst (2009), Zachman framework is easy to understand and addresses the enterprise as a whole. Since it is defined independently of tools or methodologies, any issues can be mapped against it to understand where they fit. However, due to large number of cells, it is difficult to be applied in practice.

DoDAF version 1.5, published by the US Department of Defense, provides three sets of views: operational, system and technical standards. In its evolution, DoDAF version 2.0 extends the viewpoints (previously called as views) into several perspectives, such as capability, data and information, operational, project, services, standards and systems. The project viewpoint describes the relationship between operational and capability requirements and the various projects being implemented. It also details out the dependencies among capability and operational requirements, system engineering process, systems design, and services design within the Defense Acquisition System process. The framework provides descriptions of final products, guidance and rules for consistency. Although DoDAF has a specific target, it can be extended to more general system architectures.

FEAF was developed and published by the US Federal Chief Information Officers (CIO) Council and is intended to be used by individual federal agencies. It allows flexibility in the use of methods, work products and tools. TEAF was published by Department of Treasury in July 2000 because the department was comprised of a number of offices that function as individual offices. Thus, enterprise architecture is needed to map the interrelationship among the organizations to better manage the IT resources.

TOGAF is one of the frameworks to develop enterprise architecture with detailed method and a set of supporting tools. Developed and maintained by members of The Open Group who are working within the Architecture Forum, it may be used freely by any organization wishing to develop enterprise architecture for use within that organization. The TOGAF version 1.0, developed back in 1995, was originally based on the technical architecture framework of information management which was built by the US Department of Defense (The Open Group, 2011). The following section describes how the transformation process should be executed effectively according to EA frameworks.

2.3.1 TOGAF on EA Transformation

TOGAF provides a best-practice framework for adding value and enables organizations to build workable and economic solutions which address the business issues and needs. The framework was a result of collaborative efforts of more than 300 Architecture Forum member companies. TOGAF utilization would result in consistent architecture that reflects the needs of stakeholders and considers current requirements and the perceived future needs of the business.

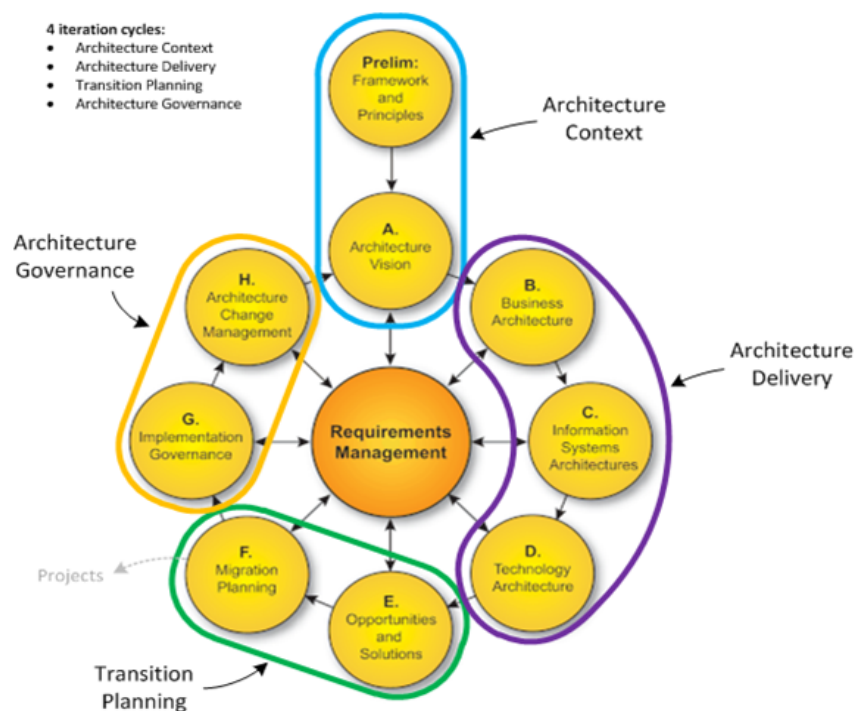


Figure 3: Architecture Development Methods of TOGAF (The Open Group, 2011)

As depicted in Figure 3, the phases within ADM are as follows:

- Preliminary phase. The preparation and initiation activities are described to prepare to meet the business objectives for a new enterprise architecture. This phase includes the definition of an organization-specific architecture framework and the definition of principles.
- Phase A: Architecture Vision. The initial phase of architecture development cycle is described. It covers information on scope definition, stakeholders' identification, architecture vision creation, and approvals collection.
- Phase B: Business Architecture. It describes the development of business architecture to support the agreed architecture vision. It covers the development of baseline business architecture, target business architecture and the analysis of the gaps of business architecture.
- Phase C: Information Systems Architecture. It describes the development of information system architecture for an architecture project, including the development of Data and Application Architectures. It covers the development of baseline information system architecture, target information system architecture and the analysis of the gaps of information system architecture.
- Phase D: Technology Architecture. It describes the development of the technology architecture for an architecture project. It covers the development of baseline technology architecture, target technology architecture and the analysis of the gaps of technology architecture.
- **Phase E:** Opportunities and Solutions. Identification of major implementation projects is conducted. The implementation projects are meant to realize the target architectures defined in previous phases.
- **Phase F:** Migration Planning. It addresses the formulation of a set of detailed sequence of transition architectures with a supporting implementation and migration plan by analyzing the costs, benefits and risks.
- Phase G: Implementation Governance. It provides the architectural oversight of the implementation and ensures that the implementation projects conform to the architecture.
- Phase H: Architecture Change Management. The establishment of the procedures for managing changes to the new architecture. It also ensures that the architecture responds to the needs of the enterprise as changes arise.
- Requirements Management. It examines the managing architecture requirements throughout the ADM. Every stage of the project should be based on validated business requirements.

With regards to EA transformation, there exist definitions of two main concept outputs: architecture roadmap and implementation and migration plan. Architecture roadmap is the lists of individual work packages in a timeline that will realize the target architecture. Implementation and migration plan is defined as schedules of projects that will realize the target architecture. Both definitions show that activities, either in the form of projects or work packages, need to be shown in a timeline view (ordered schedule) that will guide the realization of the target architecture.

The EA transformation is dealing mostly with Phase E and Phase F, as depicted in Figure 3. Phase E of TOGAF ADM, Opportunities and Solutions, describes the process of identifying the delivery vehicles, such as projects, programs, or portfolios, that will effectively deliver the target architecture which has been identified in previous phases. Since phase E focuses on how to deliver the architecture, it takes into account the complete set of gaps between the baseline and target architectures in multiple domains: business, applications and technology. Phase E is an effort to build a best-fit roadmap that is based upon the stakeholder requirements, the enterprise's business transformation readiness, identified opportunities and solutions, and identified implementation constraints. In other words, these considerations should be made available in order to perform phase E.

Step 1: Determine/confirm key corporate change attributes. The implementation factor assessment and deduction matrix is used to document factors influencing the roadmap plan. Therefore, it includes a list of factors to be considered, their description, and the implications that show the actions or constraints that must be taken into consideration when formulating the plan. In general, the factors would typically relate to risks, issues, assumptions, dependencies, actions and impacts. This also serves as the repository for architecture implementation and migration decisions.

Step 2: Determine business constraints for implementation. In this step, business drivers that would likely constrain the sequence of implementation are identified. This would include a review of the business and strategic plans as well as the enterprise architecture maturity assessment.

Step 3: Review and consolidate gap analysis results from phase B to D. This step is performed through the creation of a Consolidated Gaps, Solutions, and Dependencies matrix. The gap analysis results of phase B, C, and D must be reviewed and consolidated in a single list along with the potential solutions to the gaps and dependencies. After all of the gaps have been documented, the list must be re-organized and grouped according to the similarity. By grouping the gaps, potential solutions and dependencies can then be assessed to one or more gaps. The matrix could be used as a planning tool when creating the work packages. The dependencies identified will drive the creation of the projects and migration planning in phase E and F.

Step 4: Review consolidated requirements across related business functions. By identifying sets of requirements and integrating them into work packages, implementation of the target architecture could be made effective and efficient across the business functions which are participating in the architecture. This is important with respect to the resources provision. For example, several requirements can be resolved through the provision of a shared set of business services and information system services within a work package or project.

Step 5: Consolidate and reconcile interoperability requirements. Constraints on interoperability required by the potential set of solutions could be identified by consolidating and reviewing the architecture vision, target architectures, the implementation factor assessment and deduction matrix, and the consolidated gaps, solutions, and dependencies matrix. The focus is to minimize

interoperability conflicts and to address such conflicts in the architectures.

Step 6: Refine and validate dependencies. Initial dependencies must be refined by ensuring that all of the constraints on the implementation and migration plans are identified. Dependencies are used to determine the sequence of implementation and to identify the coordination required. Also, these dependencies could be used to determine when increments can be delivered. Addressing dependencies is fundamental for most migration planning.

Step 7: Confirm readiness and risk for business transformation. This step is done by reviewing the findings of the business transformation readiness assessment which has been conducted previously in phase A and by determining the impacts on architecture roadmap and the implementation and migration strategy. Risks associated with the transformation effort need to be identified, classified and mitigated. Finally, risks are documented together in the Consolidated Gaps, Solutions and Dependencies matrix.

Step 8: Formulate Implementation and Migration Strategy. This kind of strategy will guide the implementation of the target architecture and structure any transition architectures. Determining overall strategy to implementing solutions and/or exploiting opportunities is the first thing to conduct. The strategy could be in a form of; *greenfield*, where completely new implementation is introduced; *revolutionary*, a radical change; or *evolutionary*, a strategy of convergence like parallel running or a phased approach to introduce new capabilities. The next thing to conduct is determining the approach for the overall strategy direction that will address and mitigate the risks identified previously. The most common implementation methodologies are quick win (snapshot), achievable targets, and value chain methods. The approaches, together with the identified dependencies, are the basis for the creation of the work packages.

Step 9: Identify and group major work packages. Grouping the various activities into work packages is done by using the consolidated gaps, solutions and dependencies matrix together with the implementation factor assessment and deduction matrix. The solution column shows the recommended and proposed solution mechanism for a gap. A solution can be oriented towards a new development, based on an existing product or a solution that can be purchased. The top-level work packages should be decomposed into increments to deliver the capability increments. The work packages must be refined with respect to their business transformation issues and the strategic implementation approach. Grouping the work packages into portfolios and projects within a portfolio must be done by considering the dependencies and the strategic implementation approach.

Step 10: Identify Transition Architecture. Transition architecture is developed based upon the preferred implementation approach, the consolidated gaps, solutions and dependencies matrix, the list of projects and portfolios and the enterprise's capacity for creating and absorbing change. The transition architecture should provide measurable business value and the time-span between successive transition architecture that does not have to be the same time period.

Step 11: Create the Architecture Roadmap & Implementation and Migration Plan. Work packages and transition architectures must be consolidated to form an initial architecture roadmap. It describes a timeline of progression from the baseline architecture to the target architecture. It must show how the selection and timeline of transition architectures and work packages realize the target architecture. Implementation and migration plan must depict the necessary activities to realize the architecture roadmap. Thus, both architecture roadmap and implementation and migration plan must be aligned. The detail of implementation and migration plan must be sufficient to identify the necessary projects and resource requirements to realize the roadmap. Effective implementation and migration plan requires a clear understanding of the dependencies and life cycles of in-place solutions building blocks.

Phase F, migration planning, shows how to move from the baseline to the target architecture by finalizing a detailed implementation and migration plan, including the architecture roadmap. The objectives of migration planning phase are to finalize the architecture roadmap; to ensure the coordination between implementation and migration plan with the enterprise's approach to manage and implement change in change portfolio; and to ensure that key stakeholders understand the business values and costs of work packages and transition architectures. The steps of performing the migration planning as defined by TOGAF are summarized below:

Step 1: Confirm management framework interactions for Implementation and Migration Plan. This is about the coordination with management frameworks within the organization, such as business planning, enterprise architecture, portfolio/project management and operations management.

Step 2: Assign a business value to each work package. TOGAF provides Business Value Assessment Techniques to estimate the business value for each project. Since the research is focusing on the roadmap or transformation plan, detailed discussion on how the business values are calculated will not be covered.

Step 3: Estimate resource requirements, project timings, and availability/delivery vehicle. This step provides the initial cost estimates. Capital cost, to create the capability, and operations and maintenance costs, to run and sustain the capability, need to be detailed out.

Step 4: Prioritize the migration projects through a cost/benefit assessment and risk validation. By ascertaining the business values against the cost of delivering them, the projects could be prioritized. The agreed prioritized list will be the basis for resource allocation.

Step 5: Confirm architecture roadmap and update architecture definition document. The goal is to coordinate the development of several concurrent instances of the various architecture which results in transition architectures. TOGAF provides Transition Architecture State Evolution Table to show the proposed state of the domain architectures at various levels of details. How the table is developed and created is not the interest of the research.

Step 6: Generate the implementation and migration plan. This step brings all gathered plan

details by using the accepted planning and management techniques. Projects and activities, as well as dependencies and impact of change, are integrated into a project plan. All external **dependencies** should be included and the availability of the resources should be assessed.

Step 7: Complete the architecture development cycle and document lessons learned. This step considers all of the lessons learned during the development of the architecture and documents them by an appropriate governance process. However, the process of documenting the lessons learned is not the interest of the migration plan itself.

As one of the outputs of the migration planning phase, architecture roadmap should be able to list individual work packages (along with its business values at each stage), shown in a timeline, to show the progression and how the target architecture is realized. The implementation and migration plan provides scheduling of projects that will realize the target architecture. In order to perform its function, architecture roadmap should contain the followings:

- Work package portfolio. It describes the individual work packages, such as name, description, objectives and deliverables. Functional requirements defining the work package should also be included. The business value of the work package deliverable should be stated as well.
- Implementation factor assessment and deduction matrix. This includes the necessary information to perform related analysis; risks, issues, assumptions, dependencies, actions and inputs.
- Consolidated gaps, solutions and dependencies matrix. This covers the architecture domain, gap, potential solutions and **dependencies**.
- Transition architectures (if any). This describes the enterprise at an architecturally significant state between the baseline and target architectures. These provide interim target architecture upon which the organization can converge.
- Implementation recommendations. The information included is related to the measurement criteria for the project effectiveness, risks and issues.

Schekkerman (2008) identifies step-by-step process in developing the roadmap plan and emphasizes that the changes needed to transform from baseline architecture to target architecture cannot be achieved in a single step. It takes multiple concurrent interdependent activities and incremental builds. Thus, transformation plan is needed to understand and control this complex evolutionary process.

1. Identify gaps. Gap analysis identifies differences between two states of EA in all related EA results so that EA core team can determine what components are needed to be changed, added or eliminated to achieve the desired EA state. **Dependencies** among the developmental programs and components are considered in this process as well.
2. Implement an EA measurement program. The purpose of this step is to really measure and ensure that the progress and transformation steps are in line with the goals and objectives to achieve in a certain timeframe. It defines the economic value of EA.

3. Define and differentiate business, process, legacy, migration, and new systems. Prioritizing activities and projects is crucial to manage the migration so that the impacts of the unforeseen events on the enterprise operation efficiency could be minimized. Systems migration tables, diagrams or charts can be used to depict how systems and applications evolve between baseline and target architectures.
4. Plan the transformation. Migration to the target architecture needs to be planned to accommodate the organization's capacity to handle change. Therefore, understanding the level of effort is important to manage works according to the milestones of a scheduled migration. **Dependency** analysis helps to decide which activities should be completed at certain time period. This interdependency of system within the organization and the dependencies among projects are the main drivers to plan the sequence for solution implementation.
5. Approve, publish, and disseminate the EA results. The organization's management (organization's executives, managers and architects) should approve the overall enterprise architecture after the EA results have been verified and validated. Different EA users need different information and thus different subsets or levels of details of the EA. This information could be organized in levels of details and distributed in a tiered format corresponding to security risks because it holds extensive information about the organization.

By looking up at the steps defined by Schekkerman (2008) above, the dependencies and interdependencies between components under a certain level of EA being observed play crucial role in contributing to the migration plan development. And thus, this dependency relationship needs to be well documented.

2.3.2 Archimate - Implementation & Migration Extension

In order to put the guideline into practice, a modeling language is needed to provide a uniform representation for diagrams that describe enterprise architectures. The uniform representation offers an integrated architectural approach to explain and illustrate the multiple architecture domains together with their relations and dependencies. ArchiMate is an EA modeling language which is open and independent to enable enterprise architects to describe, analyze and visualize the relationship among business domains in an unambiguous way. It is an Open Group standard which has evolved to be fully aligned with the TOGAF® standard (The Open Group, 2012).

There are five main primary components of the ArchiMate language as defined by Iacob et al. (2012), namely framework, abstract syntax, language semantics, concrete syntax and viewpoints. The (conceptual) framework consists of rows and columns, which represent layers and aspects respectively, to classify architectural phenomena. The framework is a subset of the Zachman framework and defines how the enterprises are structured. An abstract syntax is in the form of metamodel which contains the definition of the languages. It characterizes the construct of the language and the relationship to other language constructs. Therefore, it is possible to

show the dependencies between different layers, domains and views of the enterprise architecture. This results in a coherent perspective instead of isolated diagrams.

The language semantics defines the meaning of each language construct and relationship type. The semantics must be very intuitive so that it could be understood even by someone with a novel IT familiarity. A concrete syntax is presented in terms of a visual notation. This type of syntax describes how the language constructs which are defined in the metamodel are presented graphically. Viewpoints are similar to the concept of diagram types in UML. Viewpoints are more flexible because there is no strict regulation in dividing the constructs into viewpoints. Thus, concepts could be used in several different viewpoints. The Figure 4 below illustrates how the components are related to one another.

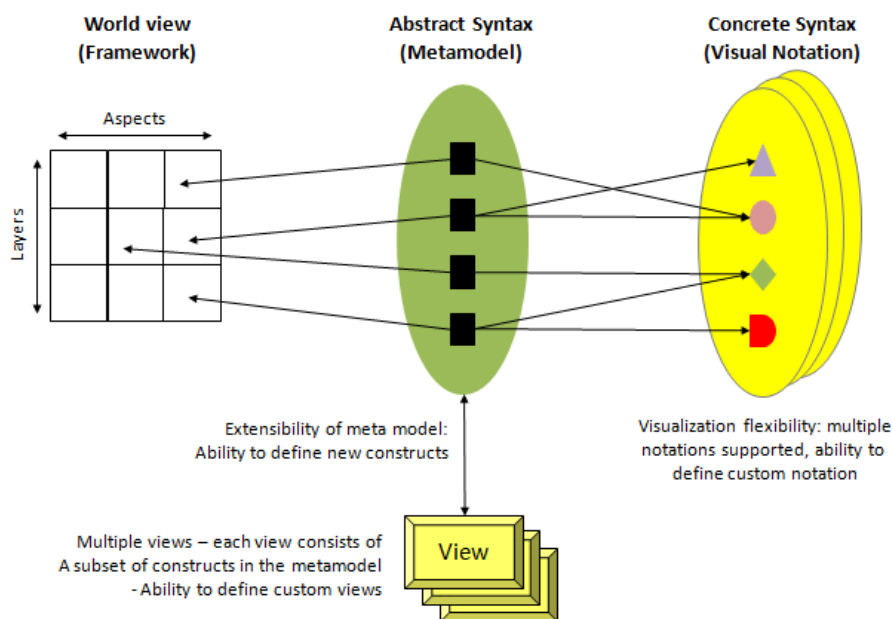


Figure 4: Components of ArchiMate Approach (Iacob et al., 2012)

In its evolution, ArchiMate includes core concepts and two extensions to accommodate the concerns of modeling concepts for the architecture artifacts description with regards to business goals, architecture principles and requirements (motivation extension) as well as projects, programs, migration, transition architecture and gaps (implementation and migration extension).

The Figure 5 below shows the metamodel of implementation and migration concepts. A work package is a collection of actions to fulfill a unique goal in a specified timeframe. It has a clear start and end date, as well as set of goals and results. The work package concept can be used to illustrate various levels of abstraction, for example projects, sub-projects, tasks within a project, programs or project portfolios. Therefore, from concept perspective, a work package is very similar to business process in a sense that it has a set of causally related tasks, aiming to produce a well-defined result. However, unlike a business process, a work package is a “one-off” process which means that its existence is obsolete once the results have been reached.

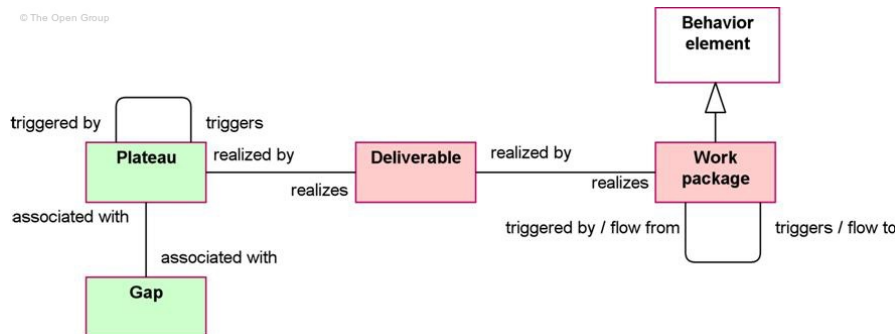


Figure 5: Implementation and Migration Extension Model (The Open Group, 2012)

For example, to illustrate the work package concept, the Figure 6 below shows a program to rationalize the application portfolio as a work package. The program consists of two projects executed consecutively, one after another, shown also in the form of work package. The first project is executed to integrate the back office system (excluding the CRM systems) and then, the second project is performed to integrate the CRM systems.

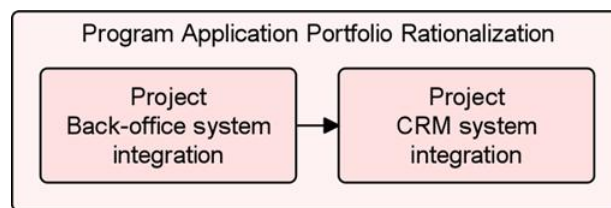


Figure 6: Work Package (The Open Group, 2012)

Deliverables are the products of work package and could be in the forms of reports, papers, services, software, physical products or intangible results like organizational change. A deliverable could be the implementation of (a part of) an architecture. They are the precisely-defined outcomes of work packages or when projects are completed. A deliverable is an architectural work product that must be specified contractually, reviewed formally, agreed and signed by stakeholders.

Plateau concept is defined as relatively stable state of the architecture that exists during a limited period of time. The concept is to support the various architectures described for different stages in time. Plateau could represent baseline architecture, target architecture and transition architectures that show the enterprise at incremental states reflecting periods of transition between the baseline and target architectures. Transition architectures enable individual work packages to be grouped into managed portfolios and programs to illustrate the business values at each stage. The order of the plateaus could be modeled by using the triggering relationships. The Figure 7 below shows an example how plateau concept is used to depict the various enterprise architecture states to model the migration from baseline to target architecture. The possible alternate transition architectures are also shown.

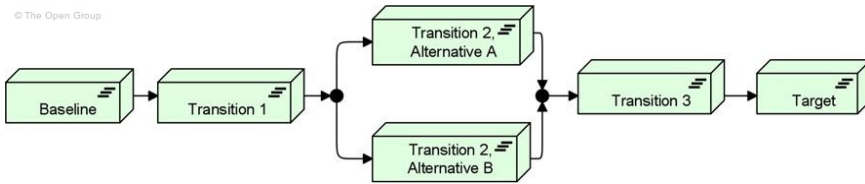


Figure 7: Plateau (The Open Group, 2012)

Gap concept is linked to two plateaus to represent the difference between the two. The plateaus could be between baseline and target architectures or between two subsequent transition architectures. Therefore, a gap is defined as an outcome of a gap analysis between two plateaus. The summary of the extension concepts with regards to implementation and migration phase could be seen in table below:

Table 2: Implementation and Migration extension Concept (The Open Group, 2012)

Concept	Definition	Notation
Work Package	A series of actions designed to accomplish a unique goal within a specified time.	
Deliverable	A precisely-defined outcome of a work package.	
Plateau	A relatively stable state of the architecture that exists during a limited period of time.	
Gap	An outcome of a gap analysis between two plateaus.	

Some specific information might need to be derived from the EA to accommodate certain interest of a user. To deliver this, a **view** concept is defined as a part of an architecture description that addresses a set of related concerns of one or more users. A view is specified by means of a viewpoint which describes how a view should be developed content-wise. Different users have different interests, operate with different concepts, have different views and need different viewpoints. In other words, a **view** is what a user sees from an EA and a **viewpoint** is from which perspective a user is looking from (Iacob et al., 2012).

According to IEEE-STD-1471-2000 (Hilliard, 2000), a view is “a representation of a whole system from the perspective of a related set of concerns” and a viewpoint is “a specification of the conventions for constructing and using a view; a pattern or template from which to develop individual views by establishing the purposes and audience for a view and the techniques for its creation and analysis”. Since viewpoints focus only on particular aspects of the architectures, they communicate only the related concerns of specific users groups. What to be included and

what to be excluded in the view depend on the relevancy and argumentation with respect to a user group's concerns.

According to TOGAF, a view is the representation of a related set of concerns, of what is seen from a viewpoint. It could be represented by a model, not necessarily be visual or graphical in nature, to depict stakeholder's interests. Thus, a viewpoint is a definition of the perspective from which a view is taken. In other words, a view is what we see, and a viewpoint is from which angle we are looking from.

With regards to implementation and migration extension, three viewpoints exist: project viewpoint, migration viewpoint and implementation and migration viewpoint. The project viewpoint is mainly used to model management of architecture change from baseline to target state enterprise architecture. The process has significant impacts on the strategy (medium or long term) as well as the subsequent decision-making process. The interested stakeholders of this project viewpoint are (operational) managers, enterprise and IT architects, employees and shareholders. Project viewpoint is suitable to see the relation between the business goals and programs or projects. It allows having an overview that all business goals are sufficiently covered by the current portfolio.

The migration viewpoint is used to model the transition of architecture change from baseline to target architecture. The user groups (stakeholders) of these migration viewpoints are enterprise architect, process architects, application architects, infrastructure architects and domain architects, employees and shareholders.

The implementation and migration viewpoint relates programs and projects to the areas of architecture that they implement. This allows modeling the programs' scope and activities in terms of the realized plateaus or affected enterprise elements which are indicated by annotating the relationships. Implementation and migration viewpoint could be related to business goals through programs and projects to parts of architecture. Therefore, analyzing the consistency between project dependencies and dependencies among plateaus or architecture elements is possible. The related stakeholders using this viewpoint are (operational) managers, enterprise and IT architects, employees and shareholders.

To characterize the content of a viewpoint, ArchiMate defines the following level of details (Iacob et al., 2012):

- **Details.** The views on this level focus on one layer and on a small portion of the architecture where many details are given. Typical stakeholders are domain architects (software engineers to design and implement a software component) and process owners (responsible for effective and efficient process execution).
- **Coherence.** The views at coherence abstraction level cover either multiple layers or multiple aspects yet in less detailed form. Multiple layers or aspects viewing allow stakeholders to focus on architecture relationships such as process-use-system or application-uses-object. Typical stakeholders are operational managers who are responsible for a collection of IT services or business processes.

- Overview. The overview abstraction level addresses both multiple layers and multiple aspects. Typical stakeholders are enterprise architects and decision makers such as CEOs and CIOs.

Based on the ArchiMate® 2.0 specification, all of the viewpoints related to the implementation and migration extension fall in “overview” level of abstraction. This means that multiple layers and multiple aspects should be covered by the views supporting the viewpoints. The general stakeholders interested in these viewpoints are mostly the enterprise architects.

Apart from the perspective of abstraction levels, the views and viewpoints are also classified from the purpose of views. There are three main types of the purposes of the views, as listed below. However, the classification does not necessarily mean that a viewpoint or view under a category cannot be used for another purpose. Some decision support viewpoints may also be used to communicate architecture to particular stakeholders as well when necessary.

- Designing. To support architects and designers in the design process from initial sketch to detailed design.
- Deciding. To assist managers in the decision making process by offering insight into cross-domain relationship, typically through projections and intersections of underlying models, but also by means of analytical techniques. Typical examples are cross reference tables, landscape maps, lists and reports.
- Informing. To inform any stakeholders about the enterprise architecture in order to gain understanding, obtain commitment and convince the opposing stakeholders. Typical examples are illustrations, animations, cartoons and flyers.

All of the three viewpoints under implementation and migration extension are supported by all of the three views’ purposes above. For example, analyzing potential overlap between project activities and dependencies would contribute to designing process in doing the transformation planning. Showing the significant consequences of the migration architecture on the strategy (project viewpoint) would provide crucial information for subsequent decision making process.

2.4 Portfolio Valuation

The work of Quartel et al. (2010) presents an IT portfolio valuation approach to deal with the paradigms that IT is more than just a cost center and that IT contributions to the business must be measured. In structuring the valuation criteria for IT, it adopts the idea of Venkatraman (1997) to identify different value centers, such as cost, service, investment and profit center. Each value center is characterized by a number of important business goals which could be prioritized. The goals are then further decomposed into several measurable key performance indicators (KPIs) which can be linked to the corresponding IT artifacts. By doing so, IT artifact can be calculated automatically from the KPIs.

Apart from the tangible benefits that could be easily quantified as KPIs, IT also constitutes less tangible benefits such as the creation of a new business capabilities or customer satisfaction.

These intangible benefits are difficult to quantify and are often neglected. By having a clear view on the value of IT rather than only its costs, an enterprise is able to decide how well IT contributes to its business goals. This ability would then enable it to make a well-balanced budgets allocation for innovation, development, maintenance and phasing out.

Quartel et al. (2010) distinguishes two types of portfolios: portfolios of IT artifacts and portfolios of IT projects. It also provides a diagram that depicts the typical ordering in the valuation of the two portfolios. First, the valuation of IT artifacts is done based on the “as-is” architecture. Depending on the outcome of this valuation, some change goals are proposed to improve the value of the IT artifacts. Then, in order to realize the change goals, projects need to be deployed. These projects are translated into a “to-be” architecture which will be used as the basis in valuating the projects in terms of the value they add to IT as compared to the “as-is” situation. Portfolios are used as the media to have a picture of the values of the subject of concerns. In other words, it could serve as a monitoring as well as a decision making means. For example, IT projects portfolios provides information about the values of IT projects. The values could consist of the costs, benefits, and risks associated to the projects. Decision making process could then be performed by utilizing the information supplied by the portfolios on deciding which projects requires extra control or needs to be phased out. Likewise, such portfolios could be applied to the gaps identified from the ADM process to give some interpretation on which gaps need to be prioritized.

Bedell’s method (Schuurman & Powell, 2008) is an interesting and useful way of measuring an IT portfolio’s values based on the business contributions. The method address the questions whether an organization should invest in information systems, on which business processes should the investment focus on, and for which activities within these processes should IT support, such as applications, be developed and improved.

The method examines the level of effectiveness of the IT supports as well as the level of their strategic importance to the business as a whole, business process and business activities. The desired ratio of the IT support is where the effectiveness is in balance with the importance. In order to calculate the ratio, the following factors must be determined and assessed: the importance of each business process to the organization (IBO), the importance of each business activity to the business process (IAB) and the effectiveness of IT supports (software applications) in supporting business activities (ESA).

Quartel et al. (2010) extends the above Bedell’s method by detailing the IBO component. The contribution of a business process to the organization can be modeled explicitly by showing the contribution of business processes to the business goals and the stakeholders (IBG). This means that IBO could be calculated from IBG and the priority of the involved goal (G). Figure 8 below shows the relation between Bedell’s method and enterprise architecture as well as the extended version with the business goals.

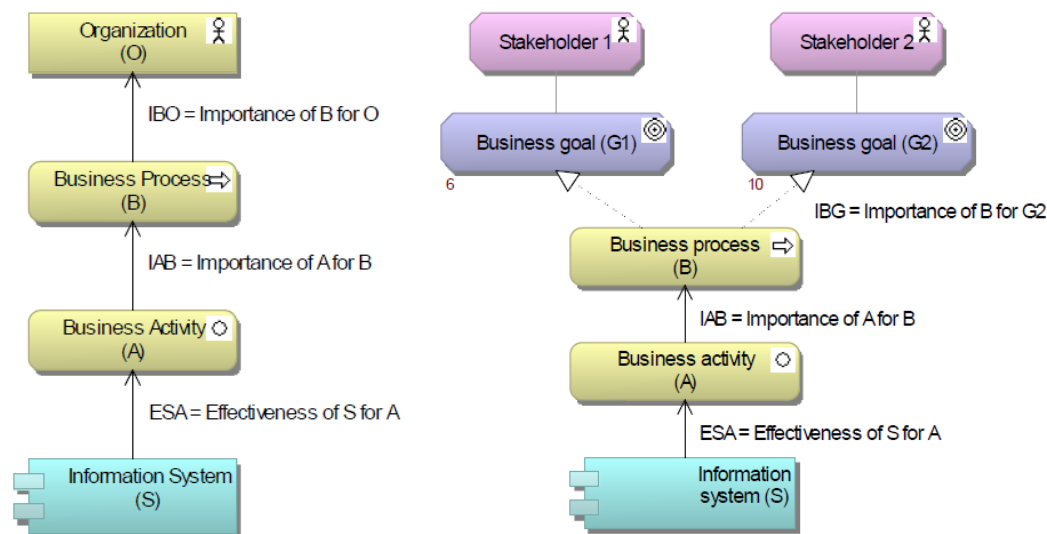


Figure 8: Bedell and Enterprise Architecture (Quartel et al., 2010)

The valuation profile as proposed by Quartel et al. (2010) borrows the idea of value center concepts from Venkatraman and adjusts it to the corresponding organization being assessed. The valuation profile could be defined by performing the following steps:

1. Define value centers. As mentioned previously, Venkatraman categorizes four types of value centers: the cost, service, investment and profit center. Additionally, budgets are allocated to each value center to distribute investment over the selected sources of business values. For example, an organization distributes different percentages of its total IT budget to different value centers: cost (40%), service (35%), investment (15%) and profit (10%). The distribution shows that the organization invests in each value center with a more focus on cost and service centers.
2. Define relevant concerns and business goals for each value center. Each value center is associated to a number of concerns and business goals. The goals are used to value the IT Artifacts and projects within a certain value center by assessing the contribution of IT to these goals. For example, service center is concerned with customer satisfaction and internal service levels. The assessment of customer satisfaction concern might reveal that customers are posting their complaints about the helpdesk. This assessment leads to the definition of a business goal "improve helpdesk". As a consequence, any IT projects that contribute to this business goal will likely to receive funding from the service center budget.
3. Define the KPIs for each business goals. The KPIs for each business goals need to be defined in order to measure the effectiveness of IT in supporting a business goal. The KPIs are obtained by decomposing iteratively a goal into sub-goals until measurable sub-goals are obtained.
4. Define the importance of business goals. The IT artifact could be related to a business goal through the architecture artifacts (business process and business activities, as shown in Figure 8) or sub-goals and requirements. IT artifact, such as software application, supports the requirements that refine the goals (goals and sub-goals).

With respects to Bedell's method, the work of Buschle & Quartel (2011) identifies the method's limitations. Bedell's method has a sort of fixed level or structure that consists of organization, business process, activities and information systems. In real life, a business process can be detailed in sub-processes or some business processes could work together to deliver a service. The method also assumes that enterprises have strict hierarchical structure and thus it neglects the situation where the same activity could be executed within multiple business processes. A one-to-one relation between business activities and ISs is strongly assumed in the Bedell's method whereby in ArchiMate language, especially in the application layer, multiple relations are supported. Another limitation is the absence of crosscut-levels relation. For example, ISs cannot be directly connected to business process; they must go through business activities.

The one-to-one relation depicts the past situation where IT systems are stand alone and are supporting a single business silos. However, today's situation requires IT systems and services to be interwoven with the business and may support different activities. Therefore an N-to-N relation is needed between business activities and IT systems. Buschle & Quartel (2011) provides generalized rules for calculating both the effectiveness as well as the importance of IT supports to businesses as depicted in the following figures.

Figure 9 shows the generalized rule for calculating effectiveness $E(S, X)$ of some IT elements S for some arbitrary architecture element X . Element $C[i]$ represent architecture elements to which S contributes directly. The effectiveness of S for these elements should have been established as input for the calculation. Elements $C[i]$ might indirectly contribute to X and that is why it is represented by dashed arrows. Therefore, the formula for calculating $E(S, X)$ requires the importance of each element $C[i]$ for X as input.

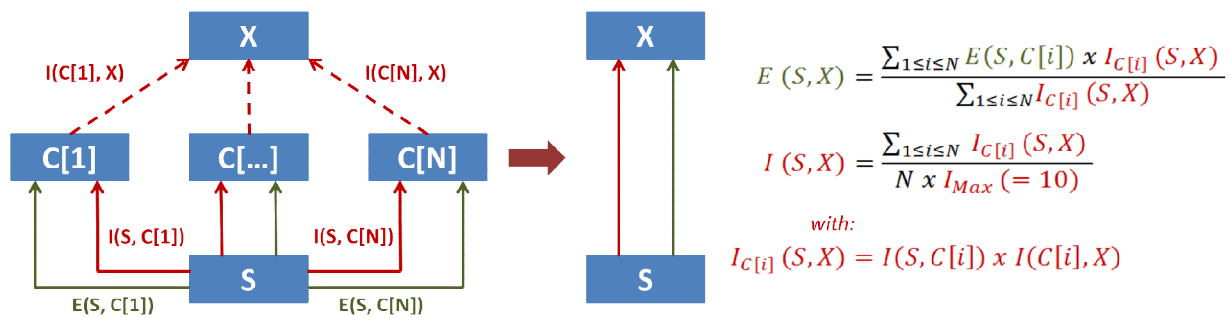


Figure 9: Calculating Effectiveness and Importance (Buschle & Quartel, 2011)

Portfolio valuation could be applied in the perspective of IT artifacts and IT projects where it aims to value IT projects according to some given criteria, compare them and decide about their acceptance or rejection. The decision making process of the project selection is the process of assessing a given set of formal project proposals with regards to one or more strategic goals and deciding to select one or more of the proposals that would optimally achieve the goals.

The work of Iacob et al. (2012a) lists down and distinguishes several selection models, based

on the IT project valuation literature, such as Financial and Economic Models; Constrained Optimization Models; Multi-criteria Decision Making Models; Checklist, Scoring models and Relevance Trees; and Architecture-based Portfolio Valuation. In addition, they also complemented the work by reviewing the strategic management literature in order to have a complete and comprehensive perspective and to avoid missing important strategy-related concepts. The most well-known theories in strategic management with regards to IT project portfolio valuation are resource-based view, dynamic capability theory and the balanced scorecard.

Among the concepts proposed by Iacob et al. (2012a) are value, risk, resource and capability. Value is defined as the relative worth, utility or importance of a core architectural element (service, product, process, application component, etc) or of a project. Risk, as defined by The Open Group (2009), is “the frequency and magnitude of loss that arises from a threat (be it human, animal, or natural event)”. Mostly, risk is calculated by multiplying the threat’s probability with the magnitude of its impact (size of value loss). Therefore, risk will be associated to an event (threat). Resource concept exists in most valuation techniques and is defined as asset owned or controlled by an individual or organization. A resource may realize a requirement which later may realize a goal. Capability is defined as the ability (of a static structure element), such as actor and application component, to employ resources to achieve some goals. This means that capability could be perceived as an abstraction of some behavior of the static structure element.

The proposed concepts of Iacob et al. (2012a) are further analyzed from the perspective of ontology-based semantics by making use of the Unified Foundational Ontology, also known as UFO (Azevedo et al., 2012). UFO has been used to analyze and interpret the semantics of the ArchiMate motivation concepts. Also, other semantic analyses, goals and business process models and role-related concepts in enterprise architecture have employed this foundational ontology. Discussion about ontological foundation could be found in Guizzardi (2005) and Guizzardi et al. (2008).

For the resource concept, Azevedo et al. (2013) suggests the addition of cardinality constraints in ArchiMate language to address the situation in which a resource is not only realized by a business actors and objects but also business roles. It argues that the lack of such constraints is being experienced by the industry and some ArchiMate tools have implemented a workaround for similar purposes. Distinction between AND and OR resource aggregations in ArchiMate could be possible to express resource optionality. By doing so, when the resource modeling element represents a pool of elements, the language would be able to define if all the resources on the pool are to be used, just one of them or any arbitrary number of them.

Likewise, Azevedo et al. (2013) also argues that the language lacks of expressiveness to state optional capabilities as well as to state if all, only one, or certain number of the capabilities associated to a behavior element are acquired or manifested. Therefore, the constraints of cardinality and the AND and OR capability aggregations could be handy to address such limitations.

2.5 Summary

This chapter presents the definition of the key terminologies which are mostly discussed throughout the research, e.g. enterprise architecture, baseline architecture, target architecture and roadmap plan. As a widely used EA framework, TOGAF defines a roadmap as the abstracted plan for the business or technology change, typically operating across various disciplines and over multiple years and thus is shown in a timeline view. Roadmap is used to describe the path of transformation process, over a certain period of time, from the baseline architecture to the target architecture.

The key user of roadmap plan, who is using it most frequently, has been identified: enterprise architect. Enterprise architect covers breadth of business and IT as compared to domain-specific architects and thus, understanding in more than one domain knowledge is required. The competencies or skills set which need to be possessed by enterprise architect could be divided into two: professional competencies and personal competencies. Enterprise architect needs to interact with other architects and stakeholders and thus personal skills such as communication skills and interpersonal skills are also needed.

Some EA frameworks have been briefly explained, and specifically emphasized on the step-by-step guideline on EA transformation process. The chapter has examined the step-by-step guideline provided by the EA frameworks in terms of its sufficiency in providing necessary directions to perform the transformation process. The ADM of TOGAF describes extensively the guideline of performing the transformation process in step-by-step activities within phase E and phase F. However, these are only general guideline where, for example, certain matrix needs to be in place in performing certain steps without describing fully how such matrix is created. This leaves an opportunity to detail out the process of performing certain step in a much detail way.

Phase E and Phase F of TOGAF ADM are very much related to project portfolio management. The main objective of these two phases is to come up with a roadmap plan which consists of list of work packages, projects or portfolios outlined in a timeline view to realize the transformation process from baseline architecture to target architecture. The relation between these two frameworks, TOGAF and project portfolio management, could be further investigated to provide more comprehensive guideline.

Portfolio valuation area is briefly discussed by reviewing the literature in this area which is related to the enterprise architecture. Two types of portfolios could be identified: portfolio of IT artifacts and portfolio of IT projects. Bedell's method was mainly referred when measuring the contribution of IT to the organization by assessing the importance and effectiveness of IT systems to the organization's goals realization. The method was further extended by Quartel et al. (2010) to address the nature of enterprise architecture where inter-relation among components (many-to-many) across the architectural layers is possible by utilizing the ArchiMate as the modeling language and Architect as the modeling tool. Several new concepts were introduced and suggested to accommodate the needs for portfolio valuation purpose, which were further assessed by Azevedo et al. (2013).

3 Current EA Supports and Limitations

This chapter discusses and summarizes the identified problems or limitations in supporting the enterprise architecture transformation, especially in the roadmap plan development. This chapter addresses research question 2 about the current support of enterprise architecture tools as well as the support limitations in analyzing and visualizing the transformation process. The chapter is organized as follows. Section 3.1 discusses how the EA transformation process is currently supported by the EA tools. Section 3.2 lists down the identified limitations of the current EA tool support, and therefore identifies potential improvements. Section 3.3 summarizes some researches or initiatives in the area of EA transformation roadmap that are available in the scientific and practical fields. Finally section 3.4 presents the summary of the chapter.

3.1 Roadmaps Visualization Support by EA Tools

Once the general principles or guidelines and the modeling language are in place, another necessary step is how to utilize those guidelines and modelings into practice. For this to be feasible and for EAs to be useful and provide business values, the development process, maintenance and implementation should be managed effectively and supported by tools. Many studies have been conducted to analyze, evaluate and compare the performance of EA tools in supporting the enterprise architecture management. Hauder et al (2013a) took the comparative study from the perspective of how well the EA tools support the metrics in enterprise architecture management. Metrics are gaining attention due to their functions in supporting the enterprise architecture analysis and evolution, such as measuring the progress of transformation and facilitating the assessment. Another study was conducted in the same year (Hauder et al., 2013b) but with different focus: analyzing task and technology characteristics for enterprise architecture management tool support. The study aimed to analyze task and technology characteristics of EA tools to handle the low utilization of the enterprises' tool solution in practice. By that, the study hoped for enhancing the performance of EA tools by incorporating unstructured information and improving the collaborative effort required to develop and maintain the EA.

Schekkerman (2011) proposed an EA tools review framework to consistently review enterprise architecture tools upon deciding which EA tool would support the organization's needs. The framework consists of two dimensions: the tool's basic functionality and the tool's utility to different professionals. The reviewer must describe the EA tools' basic functionalities in terms of how well the tool performed the various functions necessary for the EA development activities. Model development interface and analysis and manipulation are among the functionality dimensions. Model development interface is the interface used to design, build, maintain and often manipulate the models that constitute the architecture. The quality of the model development interface could be seen by how well it supports the modeling activity. Intelligent structure optimizes the utilization of limited space and supports the logical and consistent usage and navigation. Ideally, the tool must be aligned with the graphical user interface conventions and guidelines of the host operating system.

Analysis and manipulation functionality of EA tools defines the usefulness of the tools for analysis and manipulation of the developed models. The analysis supported could vary from simple, to examine how correct or complete the model is, to more sophisticated, to allow model to be evaluated in some way. It includes the ability to compare different versions or models so that the difference between current and future enterprise architectures could be identified.

The second dimension, the EA tool's utility to different professionals, captures the fitness for purpose of the tool and describes how useful the tool would be to particular professionals. There are various professionals considered in the review framework such as enterprise architects, solution architects, strategic planners/management, enterprise program managers, software architects/engineers and external partners.

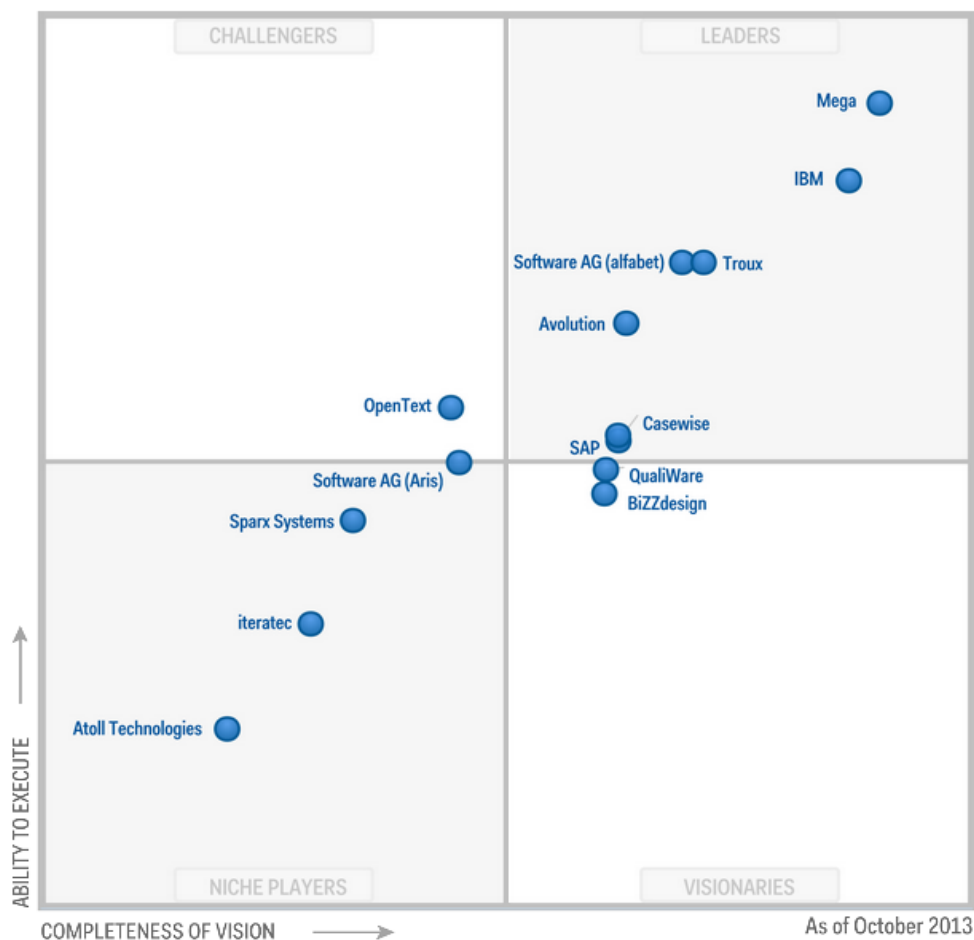


Figure 10: Gartner Magic Quadrants for EA Tools (Gartner, 2013)

Furthermore, with regards to various EA tools in the industry, Gartner (2013) released the Magic Quadrant that analyzed and compared several EA tools that are currently available in the market. The quadrant categorizes the EA tools into four quadrants; leaders, challengers, visionaries and niche players; based on the completeness of vision and ability to execute

dimensions. However, the quadrant focuses primarily on the vendor's placement in the market and not specifically on the product functionality.

According to Gartner (2013), EA tools must address various requirements of business and IT stakeholders in the organizations. Among the requirements is the decision analysis capability, such as gap analysis, impact analysis, scenario planning and system thinking. Presentation capability is also one of the requirements in which the visual or interactive visualization is in place to meet the demand of stakeholders. The Figure 10 above depicts the Gartner Magic Quadrants of the EA tools as of October 2013.

The Leaders provide mature offerings that meet market demand and demonstrate vision necessary to sustain their market position as requirements evolve. They possess a large, satisfied customers' base. A broad range of capabilities to support EA and ability to deliver these capabilities to a diverse group of stakeholders are key characteristics of the leaders. The challengers have strong ability to execute but they do not really possess a plan that will maintain a strong value proposition for new customers. The challengers could become leaders if their vision develops.

The visionaries have a thorough understanding of the necessary attributes needed yet they lack of proven capabilities to deliver against the vision. They introduce new technology, services or business models and may need to build financial strength, service and support, as well as sales and distribution channels. Expanding networks and partnerships is needed for the visionaries to further improve and become the challengers or even leaders. Meanwhile, niche players tend to have strengths in numerous aspects of EA but might be deficient in functional breadth, global presence, industry breadth or market focus. They perform well in a segment of a market where they focus on a functionality or geographic region, thus making them limited customer base.

The following sub-sections briefly discuss about the roadmap capability of EA tools in supporting the migration or transformation process. As depicted in Figure 10 above, both IBM (Rational System Architect) and Avolution (ABACUS) are under the leaders' quadrant where BiZZdesign (BiZZdesign Architect) is still in the range of the visionaries.

3.1.1 IBM - Rational System Architect

Positioned in missionary quadrant of Gartner MQ, IBM Rational System Architect establishes positive reputation in the market. It offers an enterprise planning solution to build and enhance business and enterprise architecture by providing an integrated collection of models and analytics for analysis, collaboration, and deployment of business and technology change and transformation initiatives.

In addition to the ArchiMate implementation and migration extension metamodel or concepts, Rational System Architect introduces its own concepts: encyclopedia and workspace (Owen, 2013). Encyclopedia contains diagrams (views) and definitions (concepts). By default, one encyclopedia is a namespace for all views and concepts. Encyclopedias are configured to support workspace through the system architect encyclopedia manager. Workspace is useful for

modeling architectures within specific bounds and for modeling current and future states. It is created in a tree structure so that the previous workspace could be base-lined. Thus, child workspace would inherit the views and concepts from parent workspace. The Figure 11 below shows the ArchiMate migration and implementation extension which is further extended by the IBM rational system architect.

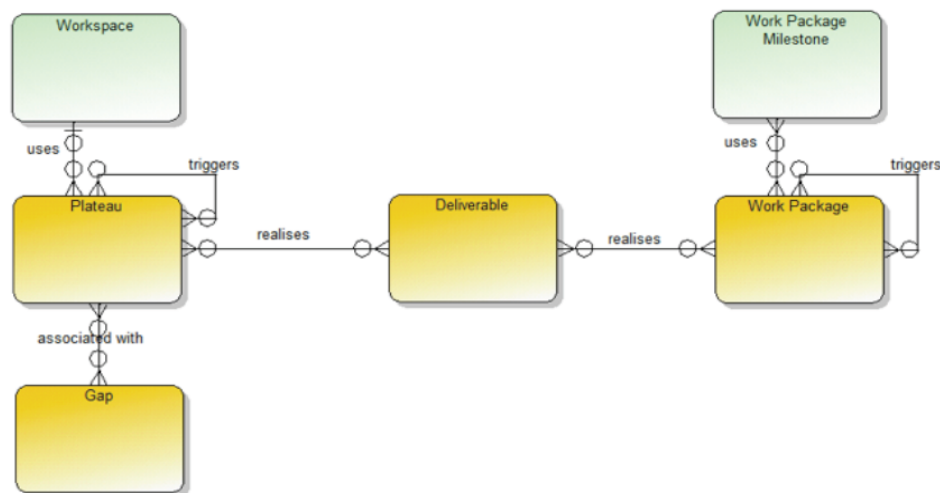


Figure 11: Additional Concepts to Migration and Implementation Extension (Owen, 2013)

As shown above, the work package has been extended to accommodate the work package milestone concept. The milestone represents an action or event marking a crucial change or stage in a work package. Each milestone possesses a date to state its end date. For each work package, numerous milestones can be assigned. An architect should identify set of threads or dimensions of life cycle of a work package, such as cost savings, resource requirements, classification etc. Color-coding could be used as indicators for the status of work package at different points in time.

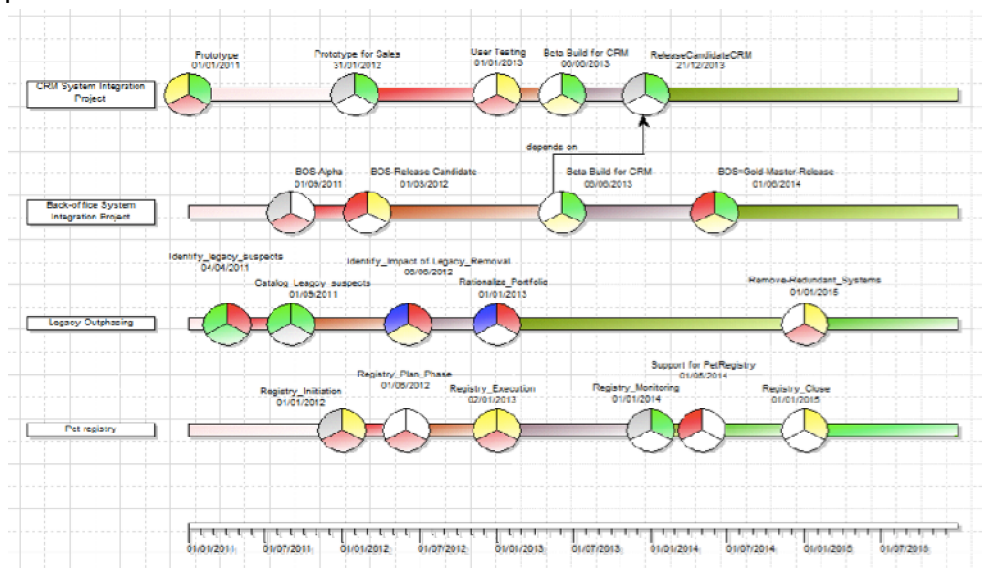


Figure 12: Milestone View Showing Work Package with Status (Owen, 2013)

Figure 12 shows a program timeline diagram and contains multiples work packages that are organized vertically and show their milestones along the horizontal access. The milestones are depicted in the form of pie charts with different color-segments to reflect the thread/dimension value selections. In addition, the timeline view could be edited with a start and end date, an interval unit and number. The diagram view, thus, only changes to represent the selected time duration.

Another addition on IBM Rational System Architect is the lifecycle states to concepts in ArchiMate in order to fully support the ability to heatmap and produce lifecycle states for architecture artifacts over time. There are two kinds of lifecycle state properties that could be added to application component concept: deployment and usage lifecycles.

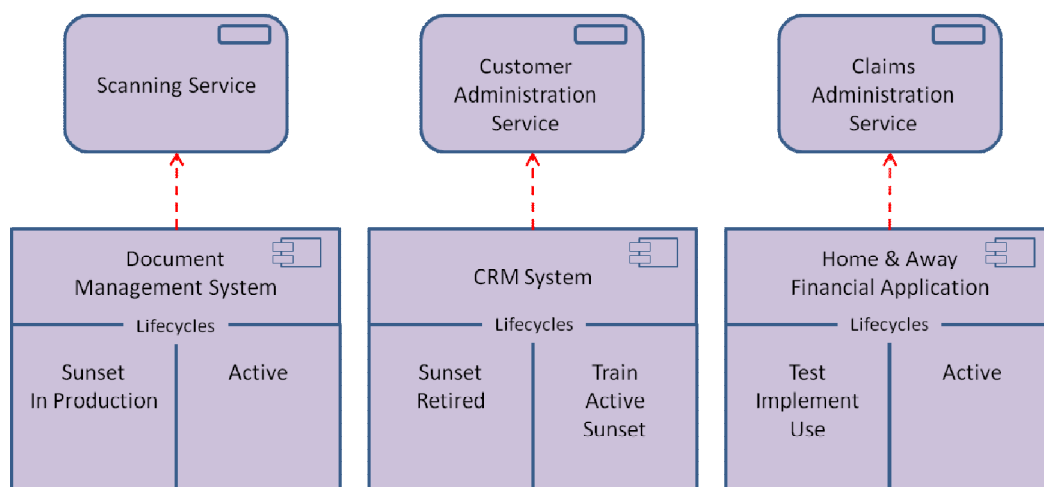


Figure 13: Additional Lifecycle States to Concepts (Owen, 2013)

In Figure 13, the left side represents the deployment lifecycles of application which corresponds to ITIL deployment lifecycle states. The names of the deployment lifecycle could be modified to align with the terminologies used in an organization. Meanwhile, the right side represents the usage lifecycles. As users interact with product or service, they continue through a series of steps called the usage lifecycle. It is used to map the users' actual usage as proposed to the deployment view of the world.

In conclusion, ArchiMate has provided the core building blocks for roadmapping. With the additional concepts introduced in IBM Rational System Architect, these combinations allow enterprise architects to model transition plans, work package and time in coherent manner.

3.1.2 Avolution - ABACUS

Avolution, with more than 12 years of experience in the EA domain, offers ABACUS as its enterprise architecture tool to the market. Categorized as leader in the Gartner MQ, ABACUS offers a comprehensive modeling solution across and between business, application, data and technology domains. It can create multiple solution alternatives and run various simulations against each alternative for metrics such as performance, cost and availability. Thus, it could

recommend the optimal path for investment, with predictive and quantitative certainty.

ABACUS implements drag-and-drop operation as a convenient way to map the information demand of visualization to an EA information model. Through this way, the existing model elements can be mapped to visual symbols on an instance of a visualization type, as a means to browse through the respective instances. A model element which has been previously linked to a visual symbol can be added to the visualization via drag-and-drop.

Hierarchy Path	Name	Implemented By/ Connections from Service Components	Stage	Business Hours	Status	Criticality	DR Provision	Business Fit	Technical Fit	1. Start Evaluation Date	2. Start Development Date	3. Begin Testing Date	4. Into Production Date	5. End of Life Date
(All)	(All)	(All)	(All)	(All)	(All)	(All)	(All)	(All)	(All)	(All)	(All)	(All)	(All)	(All)
Hierarchy Path: Applications Front-End Processing - 4 item(s)														
Applications Front-End Processing	ABACUS	Strategy	Production	9am to 5pm	Retain	3	Gold	3	3	Aug 2005			Nov 2005	Nov 2035
Applications Front-End Processing	CRM	CRM	Production		Redesign	5	Gold	-2	1	Jan 2000	Mar 2001	Jan 2002	July 2002	June 2022
Applications Front-End Processing	IVR	CRM	Production	7am to 7pm	Refresh	5	Platinum	2	-3	Jan 2005			July 2005	June 2030
Applications Front-End Processing	Website	CRM	Production	24 hour	Retain	5	Platinum	1	2	July 2008	Oct 2008	Mar 2009	July 2009	June 2029

Figure 14: Example of Application Catalogue (ABACUS, 2013)

In terms of data management support, ABACUS implements catalogue management of data. It allows the users to manage a portfolio or catalogue or list of elements and their properties natively in the tool as a tabular list of data through the standard user interface. Tabular data entry and management is fundamental to efficient management of data around objects and relations. Providing a single form or screen per object for data editing supports the data obsolescence guarantee (ABACUS, 2013). Figure 14 above shows an example of catalogue management of data which is applied in the level of applications management.

The above catalogue management of applications data could then be visualized further to ease the process of monitoring and decision making. Figure 15 below depicts the dashboard for applications. For example, the dashboards show the application roadmap displaying the life cycles or status of the applications over the timeline. The application portfolio shows the applications positioning from the perspective of technical fit and business fit. Such catalogue management could be applied to the work packages or projects. The work package dashboard would then display the work package portfolio to provide an illustration of the work packages' business value, technical risk and other possible considerations. Likewise, work package roadmap could also be visualized to provide transformation journey from the perspective of work packages/projects implementation.

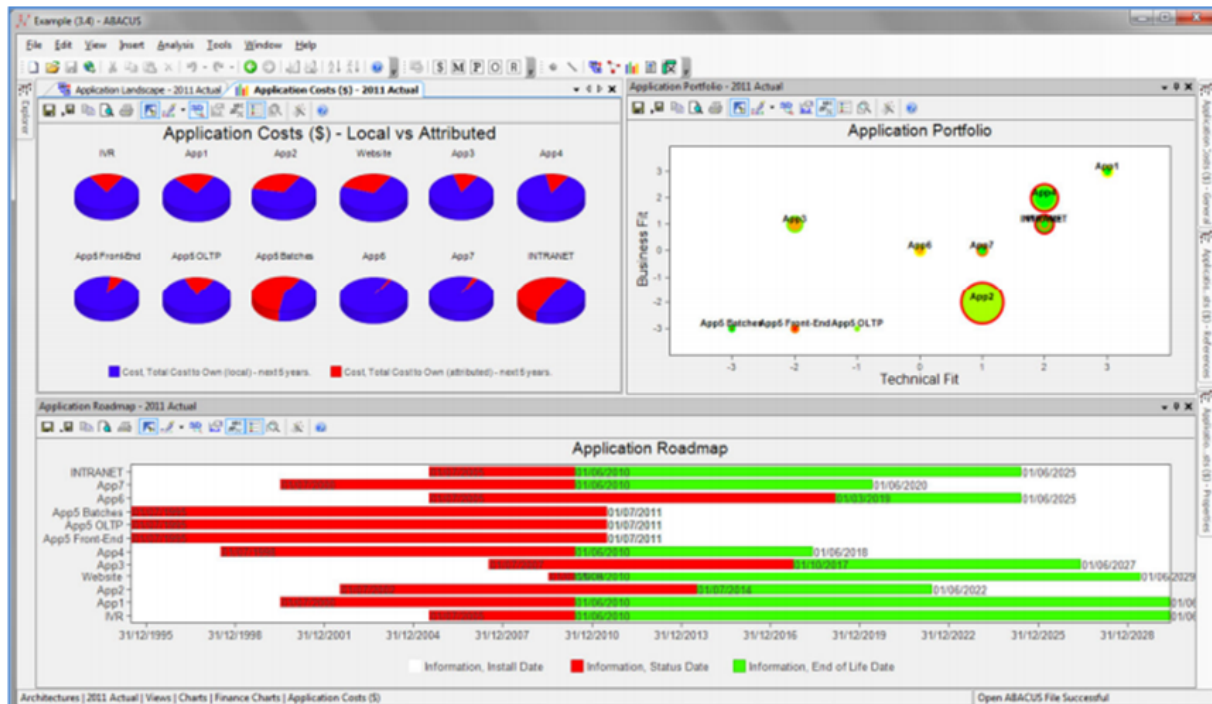


Figure 15: Application Roadmap Dashboard (ABACUS, 2013)

As noticed in the catalogue management of applications in the Figure 14, it considers the application lifecycles or status attributes and the heatmaps shown in different colors to represent different state. By implementing the catalogue management, ABACUS tries to make the roadmapping pragmatic and practical to the users. Therefore, it could be argued that ABACUS has provided sufficient and basic support to the EA transformation.

3.1.3 BiZZdesign - Architect

BiZZdesign Architect is certified by the Open Group as an ArchiMate 2.1 and TOGAF 9.1 enterprise architecture modeling tool. It is part of an integrated approach to enterprise architecture management consisting of enterprise architecture consultancy, tools and enterprise architecture training that can be customized for any organizations. BiZZdesign Architect uses its own (SQL-like) data scripting language, instead of making use of ordinary query languages like SQL or XQuery. It is a leading software tool to design and communicate architecture models and perform impact of change analysis. It is supported by the fact that BiZZdesign Architect provides broad support of data import/export interfaces as well as the color and labeling functions for diagrams.

Figure 16 shows one of the features of roadmap capability of BiZZdesign Architect. The figure depicts coloring map to show which architectural elements belong to certain state of architecture (plateau). The blue-colored elements present in both plateaus, the orange-colored elements would be removed while the green-colored elements are the newly added elements.

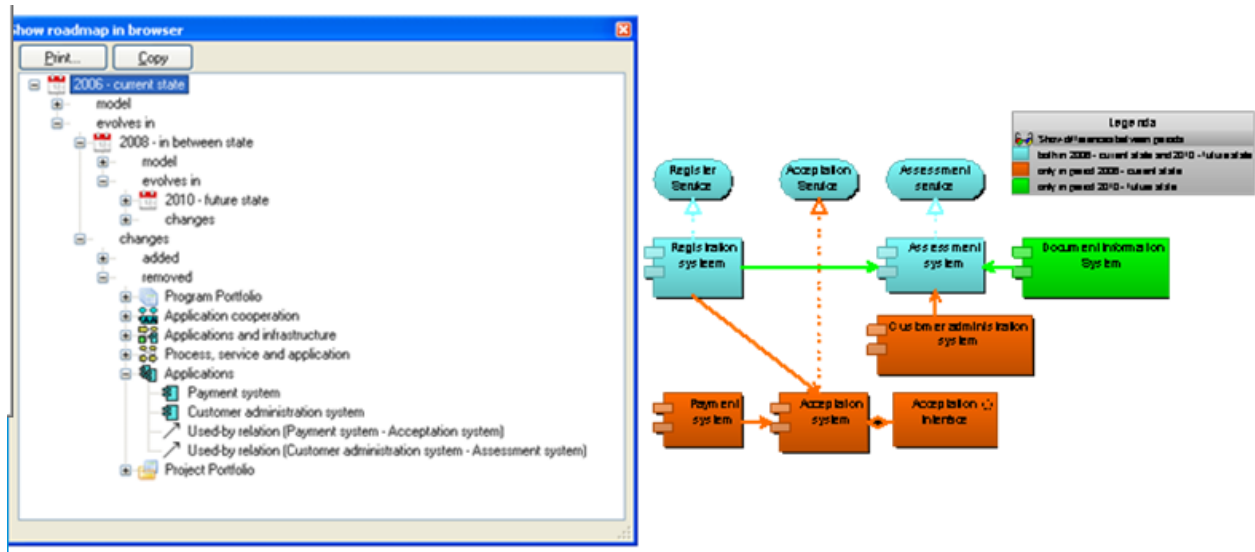


Figure 16: Roadmapping Browser of BiZZdesign Architect

There are, at least, two approaches to roadmapping performed in BiZZdesign Architect:

1. Through properties. To show that an architectural component belongs to a certain plateau, the properties of the respective component needs to be updated. The field about plateau is then to be updated with the respective plateau. This is done before the BiZZdesign Architect was upgraded to accommodate the implementation and migration extension concepts of ArchiMate language.

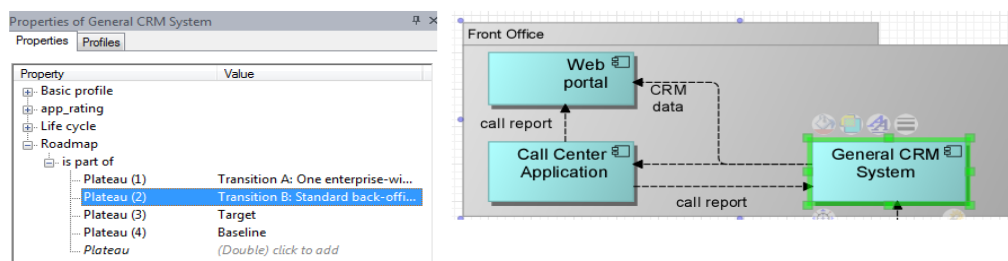


Figure 17: Plateau Assignment through Properties

2. Through plateau aggregation. Another way of setting an architectural component to be part of a certain plateau is by directly connecting the desired architecture's component to the desired plateau (drag and drop aggregation relation). This approach was made possible after the BiZZdesign Architect was upgraded to accommodate the implementation and migration extension concepts of ArchiMate language.

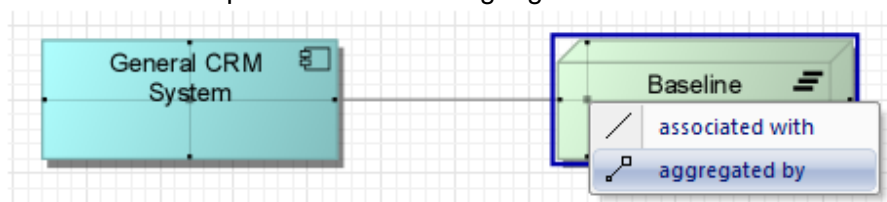


Figure 18: Plateau Assignment through Aggregation

3.2 Limitations in Roadmap Support of EA Tool

In order to illustrate the limitations or problems of EA tool in supporting the transformation process in general and roadmap plan development in particular, the research will present it in the form of case study. The presented limitations and/or problems here are also derived from the interview session conducted with the internal consultants of BiZZdesign B.V. Moreover, as stated earlier in Chapter 1, the research is conducted within BiZZdesign environment, and that being said, TOGAF, ArchiMate and BiZZdesign Architect were considered.

The case study used in this section is the ArchiSurance, a fictitious case study to illustrate the need for enterprise architecture transformation. ArchiSurance is a merger of three previously independent insurance companies, which are the original ArchiSurance (the largest of the three, offered homeowner's and travel insurance), PRO-FIT (a company specializing in car insurance), and LegallyYours (a small company specializing in legal and insurance).

As a result of the merger, the applications landscape within ArchiSurance has become scattered and increasingly complex. This situation has led to information silos which could become more and more problematic, since they refer to the same information being stored and processed at different locations. Data redundancy is one of the problems identified besides functional overlap and non-standard communication between application instances.

ArchiSurance initiated a project which aimed to clean up the application landscape. In order to do that, a systematic architecture development method is needed to guide the EA process during the migration from the "as-is" situation to the "to-be" situation. Since this research is focusing on the migration process and roadmap plan, the limitations and/or problems would be viewed from this specific phases' perspectives (Phase E - Opportunities & Solutions and Phase F - Migration Planning).

To be able to perform the migration planning, the enterprise architect needs the baseline, transition and target architectures to be in place across the business, data, application and technology infrastructure domains. The results of gap analysis are also important, identifying which components of the baseline architecture are remaining in the target architecture, which are being removed and which new components are needed to be included.

The following figures show the snapshot of the visualizations (views) created in BiZZdesign Architect with regards to the ArchiSurance case study. For this research purpose, only relevant views are shared here. According to the ArchiSurance case study, no changes in the business processes are to be performed. The transformation aimed from the merger of the three insurance companies is to clean up its application landscape. By this, it means that several applications performing the same functionality will be eliminated and replaced by one application fulfilling the needs. As shown in Figure 19 below, application architecture gap analysis has been conducted by considering the baseline architecture and target architecture of the application landscape.

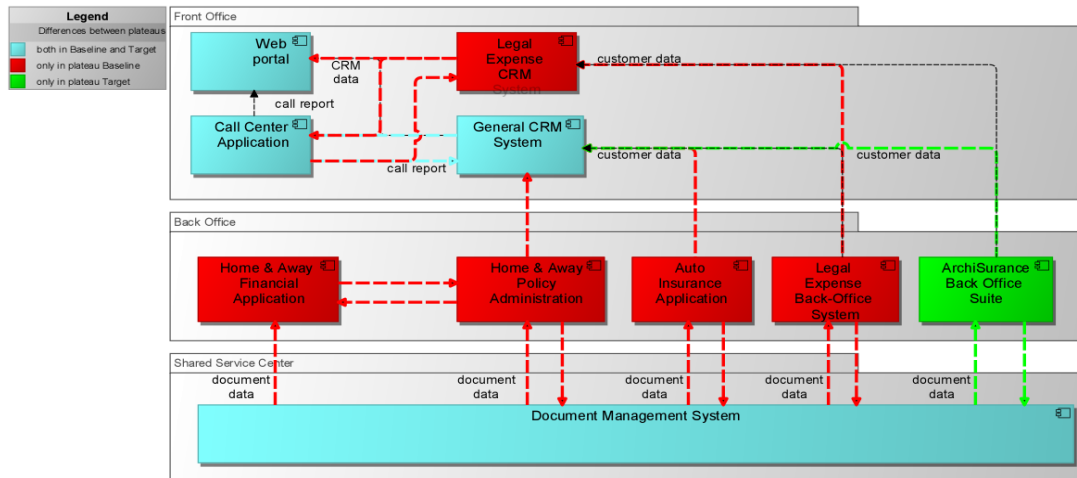


Figure 19: Gap Analysis of Application Architecture, ArchiSurance Case

The applications are supported by one or more hardware and system software, such as server. The changes in application landscape will also impact the technology or infrastructure landscape. Therefore, the Figure 20 shows the related technology architecture gap analysis. For example, Legal Expense Back Office System will be removed in the target technology since the technological tasks will be performed or supported by the ArchiSurance Back Office Suite.

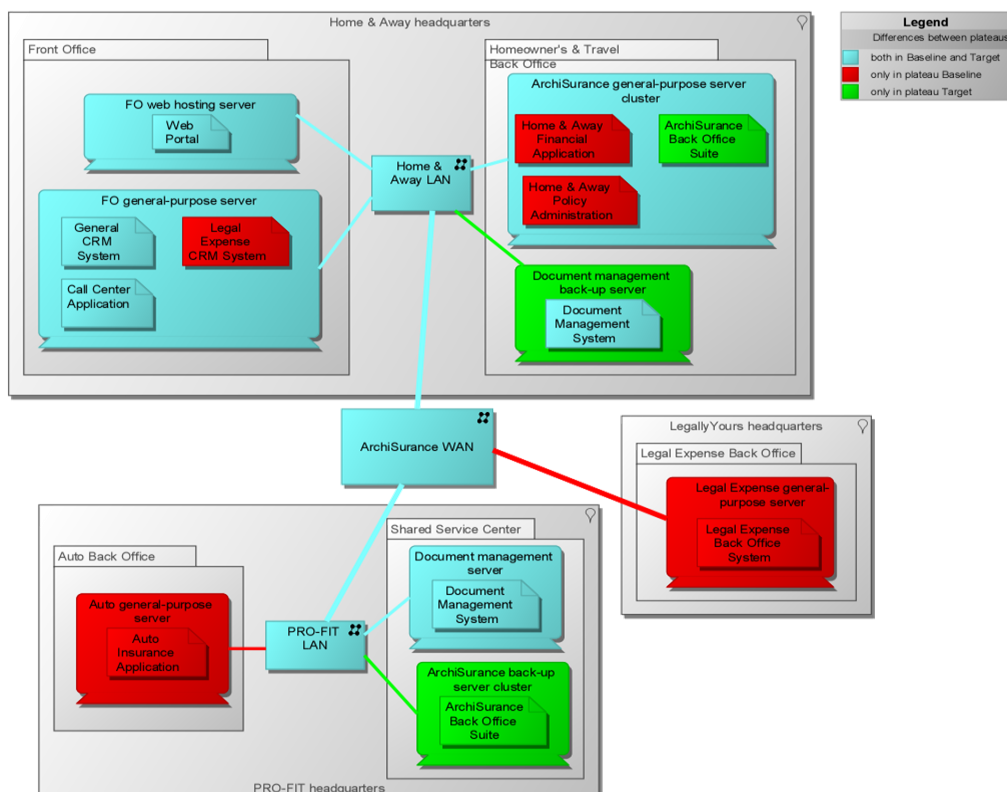


Figure 20: Gap Analysis of Technology Architecture, ArchiSurance Case

As previously explained in Chapter 2, after all of the gaps have been identified they should be consolidated. Together with the opportunities and solutions analysis, some potential work packages should be populated. Through to some further various analyses, the work packages need to be grouped into similar projects and/or programs.

Prioritization needs to take place to decide which work packages or projects are going to be conducted in which order. Concerns such as business values delivered by the projects, resources allocation and cost consumption are among the factors. Dependencies among projects are also another factor to take into account. These work packages, projects or programs are meant to be executed in order to realize the target architecture. Figure 21 shows the project context diagram supported by the BiZZdesign Architect to visualize what works packages are populated and which architectural components are influenced or affected by the work package. The view is useful to see the relations between the work package concepts to the core elements of the enterprise architecture.

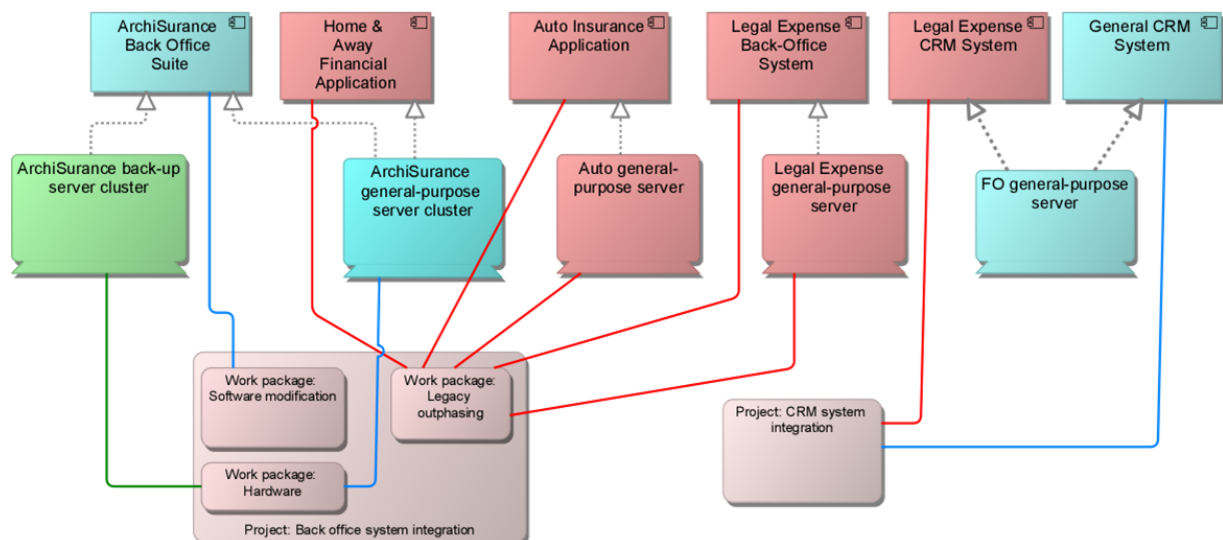


Figure 21: Project Context Diagram, ArchiSurance Case

One or more possible transformation paths could be derived from the analysis. As for ArchiSurance example, two transition architectures are identified. Transition architecture is needed to give gradual transformation so that the enterprise architecture is realized progressively, allowing the enterprise to have more control of the changes. Figure 22 shows these two possible transition architectures. Transition A is realized when the enterprise decides to integrate its CRM system while transition B is realized when standardization of back-office system is performed. These two transition architectures relate to the two populated work packages as shown in the project context diagram: back-office system integration project and CRM system integration project.

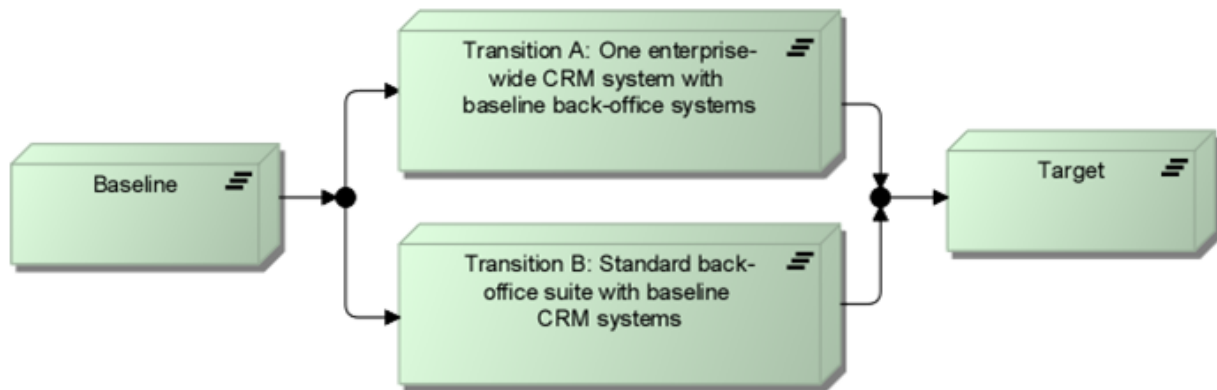


Figure 22: Possible Transformation Path, ArchiSurance Case

After the enterprise decides which transformation path it will take, then another visualization of a roadmap plan could be generated. In this example, the enterprise decides to first execute the project of integrating the back-office system. The sequence of the projects with the plateaus (baseline architecture, transition architectures and target architecture) is depicted in Figure 23 below.

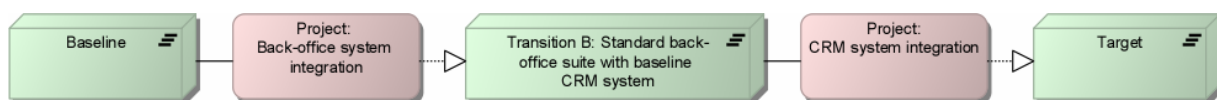


Figure 23: Sequential Roadmap Plan, ArchiSurance Case

The remainder of this section will list down scenario partitions to illustrate the limitations. This research categorizes two kinds of problems: practicality problems and guideline-related problems. Practicality problems are problems derived mainly from the daily utilization of the EA tools. These problems are the results of the consultation session conducted with the internal consultants of BiZZdesign. The session reflected the problems that are faced by the clients, or the users and in this case are the enterprise architects, especially related to the enterprise architecture transformation process and roadmap plan development.

Guideline-related problems are the problems related to the existing guidelines, provided by EA frameworks and ArchiMate. Both standards provide certain step-by-step guidelines as well as concepts in dealing with transformation process. These types of problems are identified during the literature review conducted by the author of the thesis. The standards provide general and conceptual guidelines which are sometimes too general and thus require further analysis. The author identified several guideline-related problems where further details or analyses are required or potential.

The identified practicality problems are as follows:

1. "Aggregating a relation" problem. An application sometimes will have a relationship with another application. It could be a triggering relationship or interface enabling information flow and connection between the two components. The gap functionality captures the

components that belong to a certain plateau but ignores the relationship between components. Figure 24 shows the situation, where Architecture 1 consists of application A and B, and application A triggers application B. This triggering relationship is not captured in the plateau. According to ArchiMate guideline, “aggregation” relationship should be used to indicate that a certain architectural component belongs to a particular plateau. However, we cannot aggregate a relation to a plateau. Clearly, this is a limitation in ArchiMate concept.

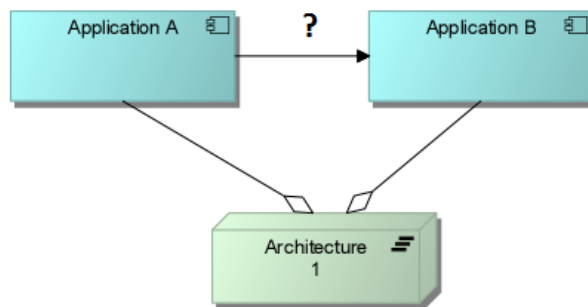


Figure 24: Aggregating a Relation Problem

2. “Updating component” problem (versioning). There would be a situation where an application is not completely removed from the Baseline architecture. Only modifications (upgrades) of the application are performed to move to the Target architecture. For example application A is upgraded to version 2 (application A_v2). This upgrade or modification will not be fully represented in the gap because the element or component is not removed nor added; only modified. Therefore, versioning of a component that belongs to two different plateaus is not well visualized. The two plateaus will show application A as if there is no update or modification occurred. In practice, architectural components (or properties of components like version numbers) are often updated in a new plateau. These components exist in both plateaus, but we do want to see that these are updated. This is currently not possible to visualize automatically.

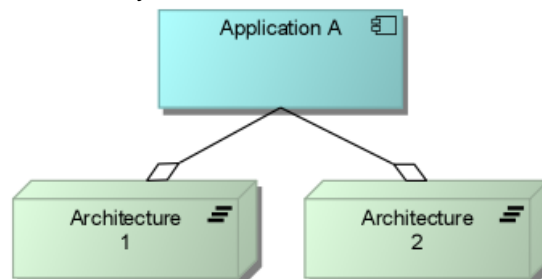


Figure 25: Updating Component Problem

3. “Date validity checking” (consistency) limitation. In Architect, an architectural element (for example an application) could be attributed startDate and endDate to depict its existence or validity. However, when performing the plateau aggregation process to indicate which components belong to which plateaus, the dates are not checked for the logical consistency. For example, an application of which the endDate has already passed might still be aggregated to a plateau of which the startDate is still not begun. This in turns would lead to planning problem in the future due to the date logical consistency.

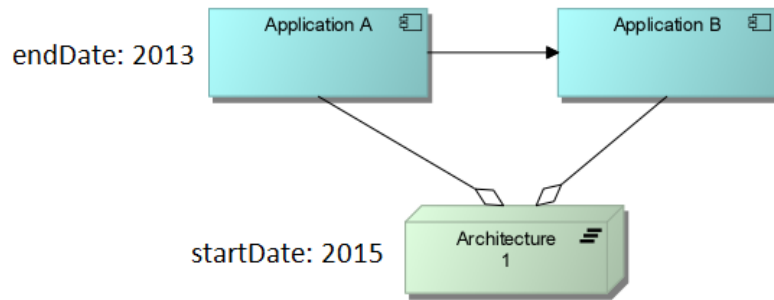


Figure 26: Date Validity Checking Limitation

4. Timeline view. Since the startDate and endDate of architectural elements are not fully used in practice, no graphical view is currently generated to illustrate the migration process, for example the milestones of necessary projects to undertake. The migration from one plateau (Baseline Architecture) to another plateau (Transition Architecture) is realized by performing necessary projects which result in removing or modifying old components and adding new components.
5. Components relationship in different plateaus. With regards to the problem number 1 above, another limitation that could be derived is the difficulty to show the relationship between applications in different plateaus. This problem arises especially when there is one application removed and a new application is added. By the current tool, we could not directly see whether the new application replaces the removed application. The new application could be possibly having no relation at all with the removed application. Take the gap analysis of the application architecture (in Figure 19) for example. By looking at the color view, we know that ArchiSurance Back Office Suite application is new in the target plateau. Legal Expense CRM application is removed in the target plateau, indicated by the red color. However, we could not directly conclude that ArchiSurance Back Office Suite application is replacing the Legal Expense CRM application. In this case, these two applications have no relation. The Legal Expense CRM application is removed because the General CRM System application will handle the tasks. Likewise, the ArchiSurance Back Office Suite application is introduced due to the removal of the other back-office applications.

Apart from the above practical problems, some potential improvements could be identified with regards to the step-by-step guidelines given by the EA framework. As we might have noticed from Chapter 2 discussion, several steps need to be performed in doing the EA transformation. However, the guideline is given in a very general description which results in several concerns. The identified guideline-related problems are as follows:

1. The implementation factor assessment and deduction matrix. It is introduced to document factors influencing the roadmap plan. The framework states that the factors would be generally related to risks, issues, assumptions, actions and impacts. A possible improvement would be to study and list down all of the related factors which need to be considered by the enterprise architect. An extensive list of factors could be populated and

stored in factors repository which later could be used as reference in assessing the roadmap plan.

2. The Consolidated Gaps, Solutions, and Dependencies matrix. It is used as a planning tool when creating the work package. The framework provides illustration of how the matrix should look like. A potential improvement would be on formal specification of how to create such matrix in the form of process guideline of what to be done to come up with work packages population.
3. Work package prioritization. The core activity of EA transformation is to prioritize the populated work packages, projects or programs and put them in a scheduled form or timeline view. This activity is heavily related to project portfolio management practice. Thus, a more detailed guideline on how to prioritize and put the projects into a scheduled roadmap plan is needed. However, this potential improvement should be carefully observed to avoid the overlapping with the project portfolio management practice.

3.3 Related Initiatives to EA Transformation Roadmap Plan

This sub-section discusses some approaches or initiatives conducted in previous researches with regards to the EA transformation and roadmap plan development. The following related works do not necessarily address the above identified problems and limitations. These works introduce other perspective of improvement with regards of roadmap plan development, such as automatic development of a roadmap plan, information model of a roadmap plan or some requirements need to be considered by EA tools in conjunction to application landscape roadmapping.

3.3.1 Visual Roadmaps for Managed Enterprise Architecture Evolution

Buckl et al. (2009) emphasized the importance of Information Technology (IT) projects which have been neglected by many EA plans while trying to project the future states of the architecture. These IT projects are actually performing the transformation of the current to the planned EA. Therefore, the research provided a viewpoint for roadmapping the development of the EA over time by introducing a (object oriented) conceptual model that explicates the demand of information for such roadmap plans.

Figure 27 shows the example of roadmap plan visualization by taking the inspiration of Gantt chart as widely used in project management. The concept of milestone is used here to indicate an important transformation event in the life cycle of a business application. Dashed, solid and dotted line styles are to represent under development, operational and in-retirement life cycle information of the application. The milestones and the lifelines are arranged alongside the temporal axis (x-axis) at the bottom of visualization.

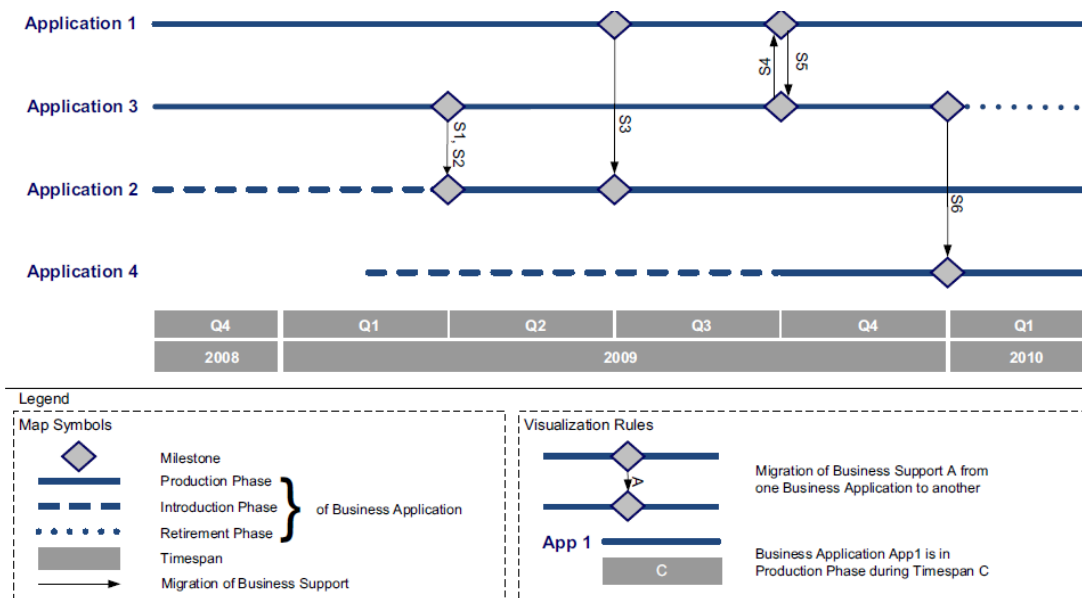


Figure 27: Business Support Migration Roadmap Plan (Buckl et al., 2009)

Figure 28 shows the information model fragment to provide the concepts necessary to describe a roadmap plan, like business applications, business support and business support migrations. Associations among the concept are defined as well in the information model to have a consistent modeling.

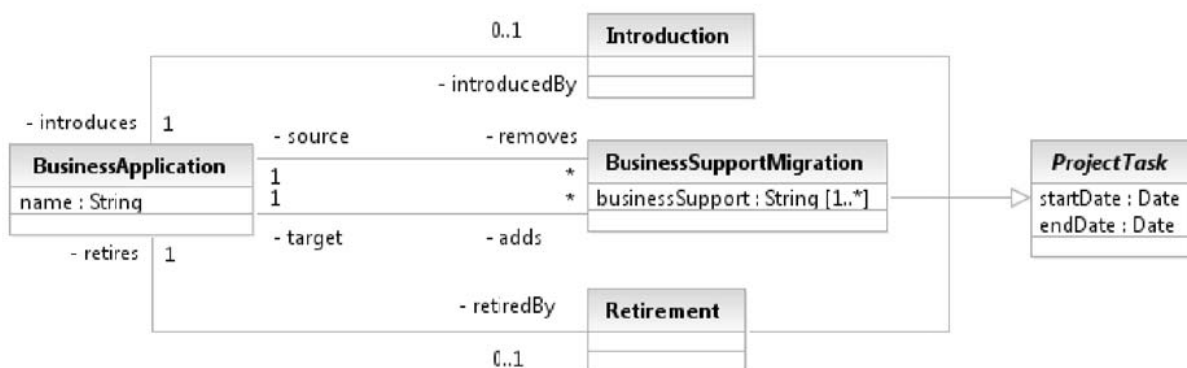


Figure 28: Information Model (Buckl et al., 2009)

The following terminologies are depicted to explain the semantics of the respective classes, attributes and associations:

- Business Application. Software system, part of business information system of an organization. It provides support for business process.
- Business Support Migration. Representing a project task of migrating the provision of a specific business support from a source business application to a target one. It is considered as completed if the **endsAt** has passed.
- Introduction. A specific type of project task introducing a distinct business application. A business application is considered in production after the date is specified in **endsAt**.

- **Project Task.** Abstract concept for different accomplishments of projects as considered in this pattern. Each project task spans a distinct period of time, between **startsAt** and **endsAt**. It indicates discrete events of change, connecting the different states of the EA to a chronological sequence.
- **Retirement.** A specific type of project task to retire a distinct business application. The business application will be in retirement after the date is specified in **startsAt**.

The limitation of the research was that there is no integrated information model employed to facilitate data collection and visualization generation. Thus, no practical validation has been undertaken to the introduced information model. A case study needs to be conducted to show the applicability of the approach. Also, it could be used to show how the EA management pattern can be applied to evolve existing EA management approaches in companies (to create a new information model).

3.3.2 Interactive Roadmap Generation

The work of Diefenthaler (2013) argued that many EA tools provide support to generate visualization of enterprise architectures at different points in time, and thus in the form of snapshot view. However, they still lack in supporting the generation of different paths of transformation from one state to another. This includes the support in sequencing and scheduling the alternative ways towards achieving the target architecture. Therefore, the goal of the work is to improve the manual creation of the roadmap generation with tool support in an interactive way by considering resources and time to enable sequencing and scheduling.

Many literature discussed how the difference between two states of enterprise architectures through gap analysis. However, the analysis is mainly on the differences of elements belonging to the states while the differences of the relationships are not taken into account. The relationship between elements can be changed over time and in returns impose constraints on the creation or deletion of elements. The research evaluated how the differences of relationships between elements in the current and target state be derived to support the roadmap generation.

As a proposed solution, planning component is introduced in the context with the EA planner as shown in Figure 29 below. The EA Planner is the user who is interacting with and using the planning component to create a roadmap. Abstract actions must be formally described in the action repository by the knowledge engineers. The formal descriptions will follow the requirements of EA planner on the changes that must be part of the roadmap and supported by the planning component. The abstract actions define the preconditions and effects of an action in general. The concrete actions relate to concrete objects represented in the model and change the state.

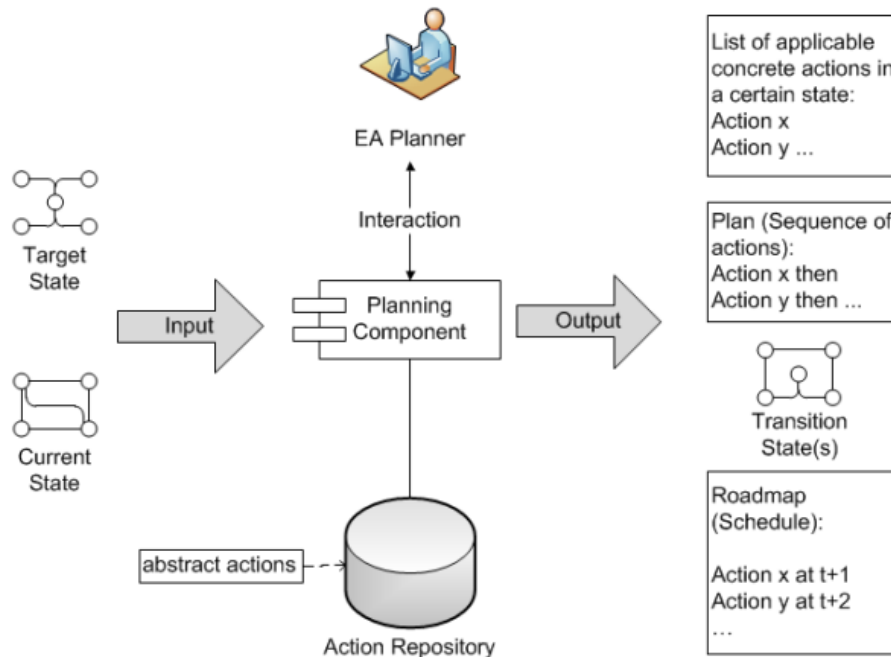


Figure 29: Planning Components in Context (Diefenthaler, 2013)

The actions repository stores knowledge of actions which need to be decided depending on the inputs. The example of an action could be to check if the plan generation is possible or not. Another example would be to check the inconsistency of the states. A plan is a sequence of concrete actions that does not take resources and time into account. A roadmap contains the sequence of actions that can be related to points in time, start and end time of actions.

The research combined the state-of-the-art in enterprise architecture planning and knowledge-based planning with artificial intelligence to automatically and interactively create the roadmap. The research listed some further improvements and directions, such as the elaboration of the method how to develop an action repository and the actions it contains. Resource and time aspects have to be more extensively considered and addressed subsequently.

3.3.3 Modeling the Transformation of Application Landscape

Hofer (2013) focused the research on application landscape transformation. The research was motivated by the fact that even though the awareness of model necessity in reaching the organization's business process transformation exists, there are no specialized modeling approaches for transformation. The aim of the research is to describe the characteristics of such a specialized modeling approach should hold. Organizations, together with their business processes and applications landscape are interwoven and thus, changes in one component will likely to affect the other components, or called co-evolution. Therefore, transformation process requires knowledge about components and the dependencies, how applications support business processes and how users work with applications. Such information cannot be gathered by measuring and automated analysis only. Observations, interviews and sometimes assumptions must take place. Models are therefore come in handy to record the knowledge

and make it accessible.

The research proposed six requirements that should be fulfilled by the modeling approaches that are used to support application landscape transformation. By application landscape, the research refers to the entirety of the business applications and their relationships to other elements. In addition, to make it more precise, only the applications that are used in the context with the human work will be considered. Thus, groups of applications that jointly or fully automatically carry out business processes are not taken into account. The six requirements for modeling approaches for transformation projects are:

- The modeling approach should make the available information manageable. Information on complex application landscapes is both incomplete and beyond comprehensibility. Therefore, a modeling approach should provide guideline on how to create and use models in such an environment.
- The modeling approach should be able to show contradictions. Applications are technical systems and information about them could be gathered and measured automatically. However, modeling in the context of application landscape is a social process because it involves interviews and observations about how people will use an application system. Various goals and even conflicting interests are inevitable and thus, the modeling approaches should consider contradictions.
- The modeling approach should be able to express how an application landscape supports business process. The model that shows the relations between business process and the supporting applications is needed for testing and effect analysis purpose.
- The modeling approach should be able to express an application landscape's dependencies even for business process that use several applications and are carried out by more than one organizational unit. Since the people using the applications are used to fragmented scope which is related only to their area, information how the application landscape and business processes work together as a whole might be misunderstood and inaccurately interpreted. The division of work results in little understanding of overall process.
- The modeling approach should be able to express dependencies between applications even if they cannot be mapped into technical interfaces. The model should consider the types of dependencies that do not correspond to any technical interface (call functions, methods, network segment, and virtual machine) and can only be recognized by analyzing the business process: dependency by time and order.
- The modeling approach should be able to express how an application landscape changes over time. Since application landscapes experience series of change until the target state is reached, it is important to know how and when changes will affect the work processes.

The research also evaluated some of the existing modeling approaches with regards to the six requirements along with the ratings where (++) means fulfilled, (+) rudimentary but insufficient solution for the requirement, (=) requirement not fulfilled but approach offers means for

enhancement and (-) means not fulfilled. The Table 3 below depicts the summary.

Table 3: Result of Modeling Approaches Evaluation (Hofer, 2013)

No	Requirements	UML	ArchiMate	EAM-Patterns	MEMO	ADOit	BEN
1	Manageable information	=	=	+	=	=	-
2	Contradictions	+	-	-	-	=	-
3	Business process support	++	+	+	+	+	-
4	Dependencies across boundaries	++	++	+	+	+	+
5	Non-technical dependencies	=	=	-	=	=	-
6	Change over time	-	+	+	=	++	+

The specific requirements for modeling approaches are needed and beneficial to improve the suitability of the approaches for transformation projects. None of the assessed modeling approaches fulfill the requirements completely. However, the research does not suggest develop a new modeling approach. The existing approaches could be complemented so that they are better suited for transformation projects.

3.4 Summary

This chapter has discussed how the roadmap plan of migration or transformation plan is supported by the EA tools. Although the research chose only three EA tools, the implication of how these tools support the roadmap plan generation could somehow represent the other existing tools. The research took two most leading EA tools in the market according to Gartner MQ 2013: ABACUS and IBM Rational System Architect. Also, BiZZdesign Architect was explored on its capability in supporting the roadmap plan generation because the research was conducted at BiZZdesign. Also, the chapter has identified some practical problems and limitations encountered by the key user in dealing with the EA tool to support the EA transformation, especially in the roadmap plan development process by using ArchiSurance case study as an example. Moreover, some challenges with regards to the guidelines provided by the EA framework have been identified.

Some initiatives or researches proposing the performance improvement of the EA tools in supporting the roadmap plan analysis and visualization are discussed. The above researches agreed on the importance of analysis and visualization of roadmap plan. Therefore, the support provided by the EA tools becomes necessary and needs special attention. The roadmap plan improvement could be initiated from different perspectives ranging from the timeline visualization of roadmap plan, interactive (and automated) roadmap plan generation upto basic requirements needed for the EA tools to accomplish. Together with the key findings from Chapter 2, the information gathered in this chapter will be used to design the proposed solution in improving the capability roadmap plan of the EA tool.

4 Addressing the Selected Problems/Limitations

After several problems and/or limitations, which are categorized as practicality- and guideline-related problems, have been identified in previous chapter, this chapter discusses the proposed artifacts of this research to address the problems. This chapter addresses research question 3 about how the development of roadmap plan could be improved in ArchiMate and Architect. The chapter is organized as follows. Section 4.1 briefly explains which of the problems have been selected to be further elaborated and addressed. Section 4.2 introduces the first artifact, which deals with the first problem: the aggregating a relation problem. Section 4.3 introduces the second artifact, in the form of step by step framework, which deals with the second problem: consolidate gaps, solutions and dependencies matrix. Finally, section 4.4 summarizes the chapter.

4.1 Selected Problems/Limitations

In order to select the previously mentioned problems, several considerations are taken into account. The first consideration is the existing research currently being conducted. This consideration is important to avoid redundant effort spent over the same or similar focus. The researches considered here are both research conducted by the internal Research and Development (R&D) team of BiZZdesign as well as the research currently being performed by other graduate interns.

The second consideration is feasibility perspective which takes into account the time limitation and additional knowledge required to further analyze and solve the problems. This factor considers the time allocated and the efforts needed towards the completion of the thesis assignment. Problems or limitations which were considered to take more time and effort outside the timeframe will be given less priority. The selection process was conducted together with the academic and industry supervisors to have a balanced thought on which problems to focus on. Academic supervisors provided insights on research trends as well as time prediction. Industry supervisor provided information about the running and existing projects currently being executed which might be related to some identified problems or limitations.

The updating component problem, which is related to versioning, is considered to require more knowledge about the existing concepts of the metamodel. Therefore, from the time allocation perspective, the problem is less feasible to be addressed. Date validity checking (consistency) limitation is given less priority because the problem is considered less urgent as compared to other problems. In the existing practice, the maturity of the users with regards to enterprise architecture transformation is still new and growing. The users are more interested, at the moment, in depicting their current position of enterprise architecture. Identifying their current business processes and visualizing and also analyzing their application and technology infrastructure are given more focus. Date validity and consistency checking during the architectural components-plateau aggregation process is very much executed during the transformation process, where an organization wants to change or transform its enterprise architecture, after the current architecture has been in place.

As for timeline view limitation, a project or research is currently being executed to provide solution. That particular research is aiming to solve or provide solution for this timeline view limitation. The expected result is to show the projects or work packages in timeline view in order to move from baseline architecture to target architecture. The shown projects would also consider the prioritization and dependencies among them so that projects scheduling is visualized. Therefore, since the timeline view limitation is currently being solved, this problem or limitation is not a candidate for further analysis in this thesis.

Components relationship in different plateaus problem is an interesting area to explore and there is currently no research or project dedicated to address this. Aggregating a relationship problem is another interesting area to explore. Therefore, there are two candidates from practicality type of problems. Both aggregating a relationship problem and component relationship in different plateaus problem are interesting candidates. However, components aggregation covers both practicality and concept perspectives. The problem deals with how ArchiMate defines the relationship between the architectural components and plateau component. The way it is defined right now creates a problem for the users because aggregating components, and most of the time huge number of architectural components, to the plateau results in an extensive effort of work. Therefore, addressing the aggregation problem is slightly more interesting.

From the guideline-related problems/limitations, work package prioritization problem is very much related to the project portfolio management areas. Even though there is no specific research or project dedicated to work package prioritization, many research has been performed in the similar area, for example project prioritization. Another research that is being done by another intern within BiZZdesign is focusing on project quantification method which will be the main input for project prioritization as well as project impact analysis to the whole enterprise architecture. That being said, this problem will not be considered as a candidate for further analysis.

Implementation factor assessment and deduction to list down and identify any possible factors that would impact or influence the migration strategy is an interesting area. However, since the factors cover wide areas and range from risks, issues, assumptions to actions, considering this type of problem would require extra effort. Furthermore, addressing this problem would result in the thesis not being focus on a scoped context. Therefore, this problem is not counted for further analysis.

This means, only the consolidated gaps, solutions and dependencies matrix is left for further exploration. The matrix plays an important activity in harvesting all of the gaps identified from previous phases and domains; business, information and technical architectures. The outcomes of the matrix are the potential solutions to handle address the gaps along with the dependencies among the solutions. The matrix is further used to identify necessary work packages needed to close the gaps in order to transform or reach the target architecture.

To make the thesis project as realistic as possible, one specific practicality problem and one specific guideline-related problem were selected. After the consideration as described above, there are two problems that would be addressed further: the aggregating a relationship problem and consolidated gaps, solutions and dependencies matrix limitation. By assessing and addressing the practicality problem, the paper intends to give practical contribution to the industry. Likewise, by assessing and addressing the guideline-related problem, the thesis intends to provide academic contribution in the area of enterprise architecture transformation.

Following Iacob et al. (2012a), there are several considerations in providing or suggesting solutions to the existing context. These considerations are meant to guide the solution process so that the proposed solutions would give effective way of addressing the problems. This being said, it would be expected that the proposed solutions would not create unnecessary nor unwanted side-effects to the existing context. The considerations are as follows:

Reuse and parsimony. The solution must consider the existing concepts and ideas already available in the context or field of study. Some concepts and ideas from previous research on similar subject, for example valuation techniques and models, would be reused whenever applicable. The number of additional concepts is kept to minimum and existing ArchiMate concepts and relationships are reused or specialized. This is to avoid inventing the whole wheel but rather applying the previously proposed concept into the applicable area that is not yet covered.

Alignment. Besides keeping the new concepts to a minimum, the proposed solution must support improvement in a way that it aligns with the existing language or framework. Unless there is a necessity of a drastic concept modification, it should be noted that the newly proposed improvement must be aligned with the current ArchiMate metamodel specification.

Ease of use. The new concepts are easy to learn, understand and use. This is to support the practicality of the solution implementation in the real case or industries. Therefore, the perspective of the users, in this case the enterprise architects, must be taken into account.

Model-based approach. The new concepts or the proposed solution should easily accommodate model-based approach. Since the architecture of the enterprise is depicted in the form of models, for example by using the ArchiMate language, then the proposed solutions would provide more meaningful insights when shown also in model-based approach.

In addition to the above mentioned considerations, The Open Group also dedicates a special section with regards to the ArchiMate extension guidelines. The ArchiMate core language contains only the fundamental concepts and relationships that are used for general enterprise architecture modeling purposes. Domain-specific purposes such as support for specific types of model analysis, support for communication of architectures, and capture the specifics of a certain application domain, for example financial sector, must also be facilitated.

The guidelines are meant to allow for such extensions, to serve more detailed and domain-specific purposes, without burdening the core with a lot of additional concepts, relationships and

notations which would be barely used by users in general. Two mechanisms are identified for such extensions in the ArchiMate as an addition to the core or can become part of the ArchiMate language.

The first mechanism is by adding attributes to the ArchiMate concepts and relationships. This is a simple way to enrich ArchiMate concepts and relationships by adding the additional information by means of a “profiling” specialization mechanism. In ArchiMate, a profile is the data structure which can be defined separately from the language yet can be dynamically coupled with concepts and relationships. The users are free to decide whether the assignment of a profile to a model element is necessary and when. Two types of profiles are distinguished: pre-defined profiles and user-defined profiles. Pre-defined profiles are profiles that have a predefined attributes structure and can be implemented beforehand in any enterprise architecture modeling tools. These are the sets of attributes for ArchiMate concepts and relationships that must be specified in order to execute common types of analysis. User-defined profiles are profiles that are newly added or defined by the users to extend the definition of ArchiMate concepts and relationships with supplementary attribute sets.

The second mechanism is through specialization of concepts and relationships. Specialization is a simple yet powerful way to define new concepts based on the existing ones. Just like in object-oriented concept of specialization, the newly defined concepts through specialization inherit the properties of the “parent” concepts but additional restrictions with respect to their use may apply. For example, some of the relationships that apply to the parent concepts might not be applicable to the specialized concepts. Likewise, specialization of relationships is also allowed with possible additional restrictions. A specialized concept or relationship is very much similar to the “stereotype” concept in UML. It provides extra flexibility as it allows the users to customize the language according to their needs and preferences while the underlying precise definition of the concepts is conserved. This means that analysis and visualization techniques developed for ArchiMate language still apply when the specialized concepts or relationships are used.

The remainder of this chapter will discuss the proposed solutions for the two selected problems with regards to the above listed recommendation principles. Each section will re-introduce the problem to give better explanation and understanding.

4.1.1 “Aggregating a relation” Problem

As briefly described in section 3.2 previously, the aggregating relation problem results from a condition that the relation between architectural elements, for example application components, is not captured by the current definition or specification of ArchiMate while describing the plateau concept. As can be seen in Figure 30 below, according to the ArchiMate specification on the cross-aspect dependencies, a plateau can only have aggregation relation with the core element or the architecture.

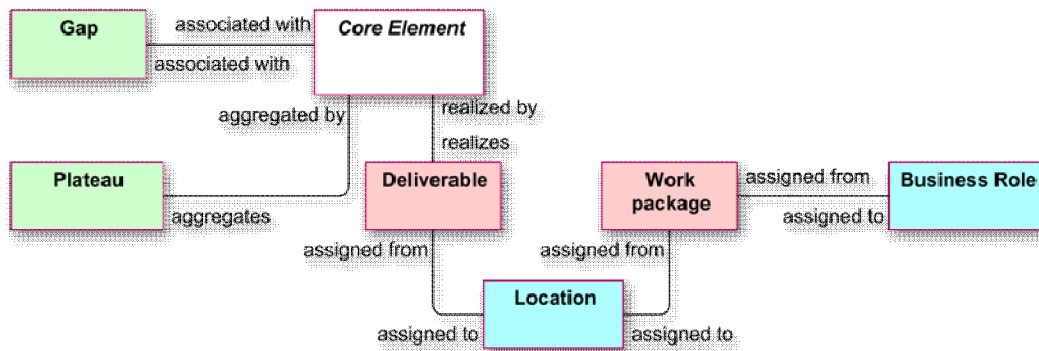


Figure 30: Relationships between Implementation & Migration Extension and the ArchiMate Core Concepts (The Open Group, 2012)

A plateau is defined as a relatively stable state of the architecture that exists during a limited period of time. It is introduced to support the way of modeling various architectures as indicated in TOGAF. Based on TOGAF's ADM cycle, phases B, C, and D each creates Baseline Architecture and Target Architecture to describe the current and desired situation of the architecture. Moreover, in phase E, Transition Architecture is defined to show the enterprise at incremental state reflecting the transition between the Baseline and Target Architectures. By having Transition Architectures, it is possible to group individual work packages and projects into managed portfolios and programs in order to illustrate the business value at each stage.

The cross-aspect dependencies of ArchiMate metamodel, as shown in Figure 30 above, results in the aggregating a relation problem. For example, an application might have a relation with another application, such as triggering relation or an interface to enable the flow of information and connection between the two components. If the two application components are valid and belong to a certain plateau, for example target architecture, then the aggregation relation between the plateau and the components could be drawn. However, the relation between the applications cannot be drawn by referring to the metamodel because aggregating a relation to a plateau is not supported.

ArchiMate categorizes three types of relationship: structural, dynamic and other (The Open Group, 2012). Structural relationships are relationships which model the structural coherence of concepts of the same or different types. The examples of structural relationships are association, access, used by, realization, assignment, aggregation and composition relationships. Dynamic relationships are used to model (temporal) dependencies between behavioral concepts. The examples of dynamic relationships are flow and triggering relationships. Other relationships that do not fall in one of these two categories are classified as other type of relationships. Other relationships include grouping, junction and specialization relationships.

The aggregation relationship falls under the structural category of relationship and aggregation is always possible between two instances of the same concepts. The aggregation relationship is

used to indicate that an object groups a number of other objects. In this plateau case, the aggregation relationship is used to indicate that a plateau groups a number of other architectural core elements. The aggregation relationship has been inspired on the aggregation relationship in Unified Modeling Language (UML). An object in aggregation relationship can be part of other aggregations. This is different from composition relationship where an object can only be part of only one composition. This is in line with the situation to describe that an architectural element can be part of various architectures (plateaus). Application A can be part of the Baseline Architecture and can be still valid and thus belong to the Target Architecture.

Alternatively, an aggregation relationship can be expressed by nesting the model elements. Nesting way of modeling is the visual way of representing the relationship between components. For some stakeholders, nesting way of modeling is easier to understand as compared to the high number relationship lines going out and coming in various architectural elements. The Figure 31 below shows the two ways to express the aggregation relationships.

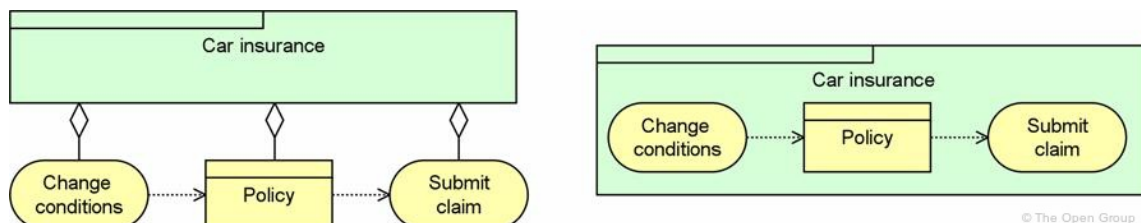


Figure 31: Aggregation and Nesting Way of Modeling (The Open Group, 2012)

However, from the perspective of the enterprise architecture tool, Architect specifically, the nesting way of representing or modeling lacks of clarity when it comes to the component to component relationship. Since nesting could also be used to mask the composition relationship, it becomes unclear or difficult to automatically distinguish between composition and aggregation relationship that is being represented by the nesting.

Moreover, the aggregation utilization within nesting representation varies depending on the enterprise architecture tool being used. For example, in BiZZdesign Architect when a component is dragged out from the nesting component (the group); the aggregation relationship that was initially established is not automatically eliminated. The enterprise architects must be very careful when using and dealing with the nesting way for it may lead to unintended misunderstanding or misinformation in the future.

4.1.2 “Consolidated Gaps, Solutions and Dependencies Matrix” Problem

As briefly described in section 3.2 previously, the Consolidated Gaps, Solutions, and Dependencies matrix is used as a planning tool when creating the work package. Although the TOGAF framework provides illustration of how the end result of the matrix should look like, it lacks of detail guidance of how to create the matrix to come up with work packages population.

The matrix is introduced and processed in phase E of ADM: Opportunities and Solutions, in the section 2.3.1. The purpose of the step 3 of phase E is to review, consolidate and integrate the gap analysis results from the Business, Information Systems, and Technology Architecture, which are performed in phase B, C and D respectively. Assessing the implications with respect to potential solutions and inter-dependencies is also the intention of the step.

The process of reviewing gap analysis and determining dependencies, as recommended in TOGAF, could be done by using sets of views such as the Business Interaction matrix, the Data Entity/Business Function matrix, and the Application/Function matrix to completely relate architectural elements from different architectural domains. By assessing those views, tracing the relations among components across multiple domains (layers) of architecture could be done and therefore, dependencies could be traced too. For example, when there is a component in a single architectural domain is affected in the gap analysis results, then the dependencies could be monitored to show which other architectural elements in other architectural domains will be also affected.

Furthermore, TOGAF ADM cycle considers and assumes that the following inputs are available in order to perform phase E. The inputs are categorized into three kinds of inputs: reference materials external to the enterprise, non-architectural inputs and architectural inputs (The Open Group, 2011).

As the reference material external to the enterprise, there are two inputs: architecture reference materials and product information. Architecture reference materials could be gathered from the architecture repository which acts as a holding area for all architecture-related projects within the enterprise. This repository allows projects to manage the deliverables, locate the re-usable assets and publish outputs to stakeholders and other interested parties.

Non-architectural inputs could be in the form of request for architecture work, capability assessment, communication plan and planning methodologies. The request for architecture work is the output of Preliminary phase as a result of approved architecture change request. The request document is at a high level and is sent from the sponsoring organization to the architecture organization to trigger the start of an architecture development cycle. Communication plan is important because enterprise architecture contains large volumes of complex and inter-dependent information. Thus, it is a key success factor to communicate effectively the targeted information to the right stakeholders at the right time.

Among the various architectural inputs, the critical inputs are the draft architecture definition document, the draft architecture requirements specification, and the candidate architecture roadmap components from phase B, C, and D. The draft architecture definition document includes the baseline and target architectures of business, data, application and technology domains. This document is the deliverable container for the core architectural artifacts created during a project and for important related information. This document covers all architecture domains (business, data, application and technology) and examines all relevant states of the

architecture (baseline, transition and target). It takes into account the works done in previous phases of phase B, C, and D. Thus, it includes the baseline business architecture, target business architecture, baseline data architecture, target data architecture, baseline application architecture, target application architecture, baseline technology architecture and target technology architecture. The baseline and target business architectures are resulted from phase B: Business Architecture. The baseline and target data and application architectures are the results of phase C: Information Systems Architecture. The baseline and target technology architectures are the results of phase D: Technology Architecture.

The draft architecture requirements specification provides a set of quantitative statements that outline what an implementation project must do in order to comply with the architecture. In each of phases B, C and D, since the baseline architecture and target architecture of the corresponding domains have been identified, gaps analyses are as well conducted. This results in the list of gaps of the business, data, application and technology architectures. Therefore, it is assumed that the lists of gaps for various architecture domains (business, information system, and technology architectures) are readily available.

By having all of these information as inputs, the purpose of the step 3 of phase E is to review, consolidate and integrate the gap analysis results from the Business, Information Systems, and Technology Architecture, which are performed in phase B, C and D respectively. Consolidating means bringing together (separate parts) into a single or unified whole of unit. The list of the gaps resulted from phase B, C and D are still separated. Thus, the inter-dependencies among the gaps across various domains need to be reviewed and assessed in order to come up with list consolidated gaps.

When the gaps have been consolidated, it would then be easier and meaningful to make some sort of gaps valuation in order to prioritize the gaps. In order to reach the target architecture, work packages in the form of projects need to be implemented which deliver several deliverables (reports, papers, services, software, physical products, intangible results such as organizational change and the implementation of a part of architecture).

Although the importance of having a consolidated gaps, solutions and dependencies matrix has been identified in TOGAF, a more detail step by step guideline is needed on how to come up with such matrix. It is therefore the main challenge of the second selected problem of this thesis research: to provide a guideline to develop consolidated gaps, solutions and dependencies matrix. Valuating the gaps is one of the objectives while developing the matrix, which later could be used as the foundation to make gaps portfolio in order to prioritize which gaps are needed to be closed first in order to realize the target architecture.

4.2 Solution to “Aggregating a Relation” Problem

Based on the analysis on the literature regarding the enterprise architecture transformation concepts definition as well as guideline provided by the enterprise architecture frameworks, enterprise architecture modeling language and enterprise architecture tool, the following

solutions are proposed. These solutions will mainly be focusing on the aggregating a relation problem.

The first solution is the main proposed suggestion to address the selected problem. It is dealing with the conceptual concern of plateau aggregation. Therefore, this solution is mostly related to the ArchiMate modeling language which provides conceptual guidance to the enterprise architecture. The other three solutions are proposed to improve the practicality or pragmatic perspective in helping the users in dealing with the enterprise architecture transformation process. Thus, these solutions are mostly related and specifically dedicated to BiZZdesign Architect.

4.2.1 Conceptual Solution

S1. Plateau concept extension to aggregate relations between core elements

The conceptual problem of the aggregating a relation problem is the limited definition of plateau concept which explicitly defines that only architectural component could be aggregated to a certain plateau. This limits the fact that relations between components could not be aggregated and thus could not be captured as part of a certain plateau. The first solution is to extend the metamodel definition of plateau that it could also aggregate the relation between the core elements.

This solution is proposed as an improvement to the ArchiMate as the modeling language of enterprise architecture. By referring back to the definition of a plateau, which is as a relatively stable state of the architecture that exists during a limited period of time, the concept of plateau is mainly about the architecture. The architecture itself is developed by collection of core elements which are inter-related one to the other. Therefore, it is understandable that the concept of plateau is then depicted as the collection of core elements represented through the aggregation relationship between the plateau and the core elements.

However, it should be noted that architecture also includes the relationship among the core elements within the architecture. Therefore, the fact that a plateau concept also includes the relationship among the core elements that belong to that specific plateau should also be depicted. This is necessary because the description of a state of architecture contains all relevant core elements, including their properties, and their relations to each other.

In order to accommodate this solution, several possible discussions are worth to consider resulting in a more concrete solution. The first discussion is about the relevance of having a “plateau” concept in the ArchiMate. Plateau is defined as the state of the architecture. It could be used to show the baseline state, transition state or target state of the architecture. With that in mind, ArchiMate should define a concept of “architecture” instead of a “plateau” concept. This would make more sense to have a concept of “architecture” that aggregates all of the core elements. The plateau could then be defined as the state attribute of the “architecture”. Since alignment is one of the concerns when deriving solutions to problem, as previously described in section 4.1, this alternative discussion is not further assessed and considered in this thesis.

Making use of “nesting” idea in the ArchiMate is another interesting discussion in representing plateau graphically. In ArchiMate, nesting is seen as another way of representing the relationship among components graphically. Graphical nesting provides representation of a structure which is easier to understand by the reader because it simplifies the view by removing the relationship between the bigger component concepts with other smaller components concept underneath it. As previously described in section 4.1.1, the limitation of graphical nesting representation is that it does not define explicitly what type of relationship that it represents. Nesting could be the masking of composition or aggregation relationship. Since plateau concept definition explicitly shows aggregation relationship, in depicting that certain core components belong to a certain plateau, then the possibility of using graphical nesting representation to extend the definition of plateau concept is not feasible.

Alternative conceptual solutions.

Based on the previous discussion and fundamental description of basic concept of plateau and aggregation relationship, three alternative solutions are proposed to extend the definition of plateau concept in ArchiMate.

1. Adding relations aggregation to plateau concept.

The first solution is to simply make the relations aggregation possible in the plateau concept. In ArchiMate, relationships and architectural elements (core concepts) are two different things. As previously described in literature review section regarding the ArchiMate modeling language, core concepts would fall into one of the three main types of elements: active structure elements, behavior elements, and passive structure elements. The figure below shows the possible relationships among the concepts in ArchiMate.

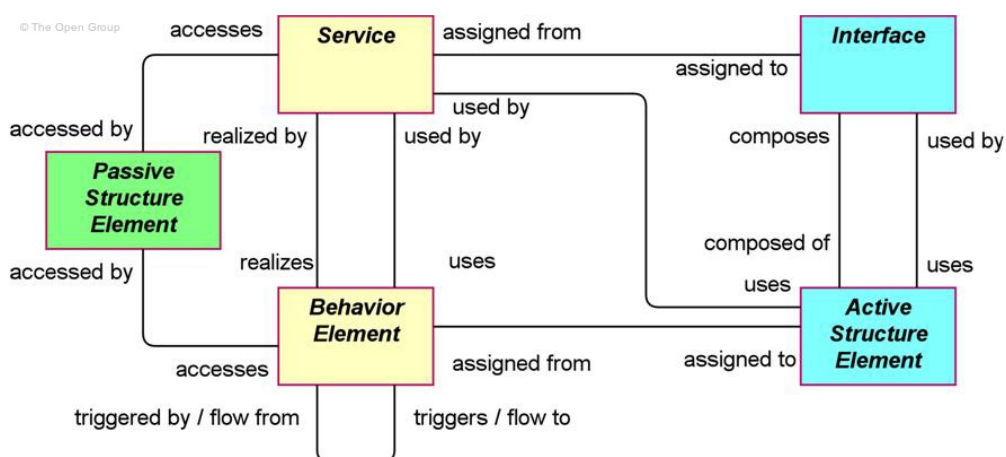

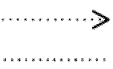






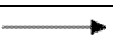





Figure 32: Generic Metamodel: The Core Concepts of ArchiMate (The Open Group, 2012)

Set of relationships have also been defined in ArchiMate, which could be classified as *structural*, *dynamic* and *other* types of relationships. *Structural* relationship models the structural coherence of concepts of the same or different types. Composition, aggregation, assignment, realization, used by, access and association relationships are under structural relationship.

Dynamic relationship models (temporal) dependencies between behavioral concepts. Triggering and flow relationships are examples of dynamic relationship. *Other* relationship is used to model the types of relationships that do not fall under structural and dynamic relationships. The examples of this type of relationship are grouping, junction and specialization relationships. The table below summarizes the core sets of relationships defined in ArchiMate modeling language.

Table 4: ArchiMate Relationships (The Open Group, 2012)

Structural Relationships		Notation
Association	Association models a relationship between objects that is not covered by another, more specific relationship.	
Access	The access relationship models the access of behavioral concepts to business or data objects.	
Used by	The used by relationship models the use of services by processes, functions, or interactions and the access to interfaces by roles, components, or collaborations.	
Realization	The realization relationship links a logical entity with a more concrete entity that realizes it.	
Assignment	The assignment relationship links units of behavior with active elements (e.g., roles, components) that perform them, or roles with actors that fulfill them.	
Aggregation	The aggregation relationship indicates that an object groups a number of other objects.	
Composition	The composition relationship indicates that an object is composed of one or more other objects.	
Dynamic Relationships		Notation
Flow	The flow relationship describes the exchange or transfer of, for example, information or value between processes, function, interactions, and events.	
Triggering	The triggering relationship describes the temporal or causal relationships between processes, functions, interactions, and events.	
Other Relationships		Notation
Grouping	The grouping relationship indicates that objects, of the same type or different types, belong together based on some common characteristic.	
Junction	A junction is used to connect relationships of the same type.	
Specialization	The specialization relationship indicates that an object is a specialization of another object.	

The following figure depicts the extension of the plateau concept definition in ArchiMate by accommodating the aggregation of relation between components to the plateau. As shown by the figure, the proposed solution expands the existing metamodel in two things. The first modification is on the self-referential relationship that goes out and into the core elements. The self-referential relationship is to show the relation between two architectural components. The possible types of relationships among the core concepts are previously shown in Figure 32. The second modification is on the aggregation relationship from the self-referential relationship to the plateau concept. This additional aggregation line would allow the metamodel to also include the relation between architectural components as part of the plateau.

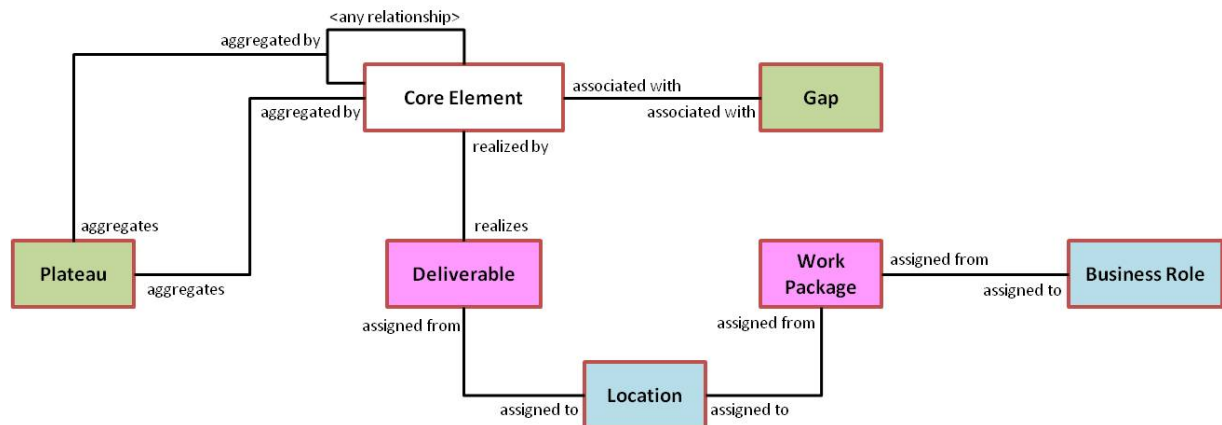


Figure 33: Extension of Relationships between Implementation & Migration Extension and the ArchiMate Core Concepts

The types of relationships among the core concepts that could be aggregated to the plateau concepts include all of the relationships that are displayed in Figure 32. For example, the following relationship types are also covered by this aggregation extension.

1. Composition relationship. It indicates that an object is composed of one or more other objects. An object can be part of only one composition. In composition relationship, it defines all or nothing state which means that an object which is composed of other objects can only exist if all sub-composed objects exist. When the big object is deleted (no longer valid), so are the sub-composed objects. Likewise, if one of the sub-composed objects is deleted (no longer valid), then the big group of composed object is also not valid. In the context of aggregation extension in plateau, composition relationship should also be aggregated to the plateau. The composition structure of components within a plateau might be modified, and thus is different, in another plateau. Therefore, even though the super object (Web portal) and the lower objects (intranet portal and extranet portal) are valid in the period of the plateau, the composition relationship among them still needs to be aggregated. This is because not only does the composition relationship hold the 'all or nothing' characteristics, but also the composition structure might change between plateaus.

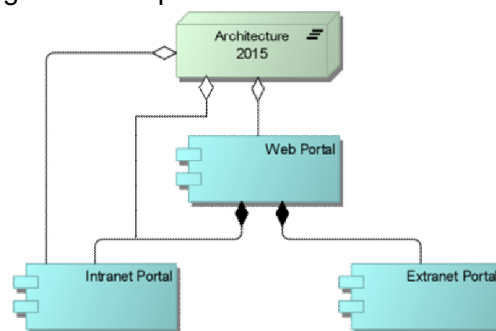


Figure 34: Composition relationship in aggregation extension.

2. Junction relationship. This type of relationship is used to connect dynamic relationship of the same types. A junction is used in a number of situations to connect dynamic (triggering or flow) relationship of the same type, for example, to indicate splits or joins. Because of this reason, a junction has an explicit component source with a target component as the decision making point. The junction point (represented as a small dot) is considered as a concept itself. When acting as a target component, it receives certain values or conditions to be analyzed. When acting as a source component, it provides values to the next target component. The relationship to be aggregated to a plateau requires both explicit source and target components. A logical reasoning behind this is to have a final target component. For example, a flow relationship requires both a clear source and target components. With respect to this, junction relationship could be included in the types of relationships that could be aggregated to plateau concept, as long as the final target component (at least one) is also aggregated to the plateau. The junction point relationship, aggregated to the plateau, represents the incoming relationship from the source component and the outgoing relationship to the target component.

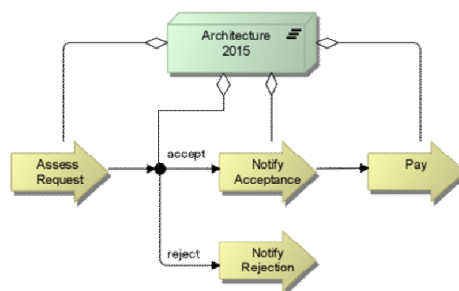


Figure 35: Junction relationship in aggregation extension

3. Specialization relationship. This type of relationship indicates that an object is a specialization of another object. The specialization/generalization relationship can relate any instance of a concept with another instance of the same concept. In the context of aggregation extension in plateau, specialization relationship should also be aggregated to the plateau. The specialization structure of components within a plateau might be modified, and thus is different, in another plateau. Therefore, even though the super object (Take out insurance) and the lower object (Take out travel insurance) are valid in the period of the plateau, the specialization relationship among them still needs to be aggregated.

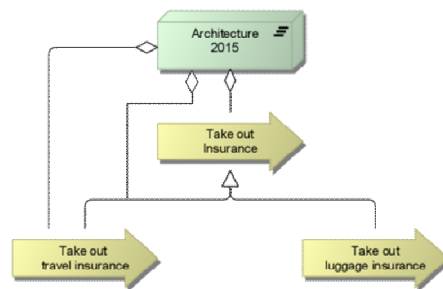


Figure 36: Specialization relationship in aggregation extension

The implementation of the proposed plateau concept extension is shown in figure below. Architecture 2015 plateau is the enterprise architecture state in the year of 2015 and thus, it is a future or target state of the architecture. Web portal application components and Call center application component belong to Architecture 2015 plateau. This is shown by the aggregation relationship from the components to the plateau. The relation between call center application and web portal application, in this case the data flow relationship, also belongs to the Architecture 2015 plateau. This is shown by the aggregation relationship from the relation to the plateau.

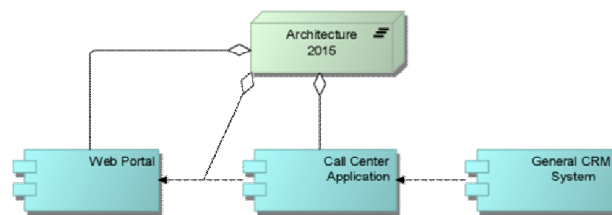


Figure 37: Example of aggregating a relation between components

The inclusion of relation between components to be aggregated to plateau concept should take several concerns into account. The **first concern** of aggregation extension to plateau concept is the target and source components. In general, for a relationship to be aggregated to a certain plateau, it is required that both source component and target component are also valid and aggregated to that certain plateau. The following scenarios explain what is possible and what is not possible with regards to this concern.

1. Both source and target components are valid to a certain plateau. Here, aggregating a relationship to plateau is possible and allowed because the plateau acknowledges both the source and target components of the relationship.

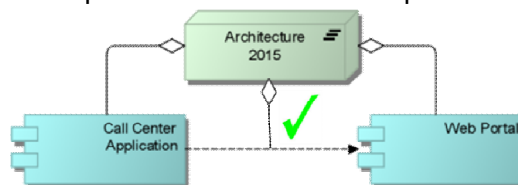


Figure 38: Scenario 1 – Target and source components

2. Source component is valid but target component is not valid to a certain plateau. Here, aggregating a relationship is not possible and not allowed because the plateau only knows the source component. Even though the relationship itself knows the source and target components by storing this information on its attributes, the plateau does not know the target component. This happens because only the source component is valid and aggregated to the plateau. If aggregating such condition is allowed, then the logic of the relationship from the perspective of the plateau would not be correct. Therefore, it is considered not possible and not allowed to aggregate a relationship where only the source component is valid to the plateau.

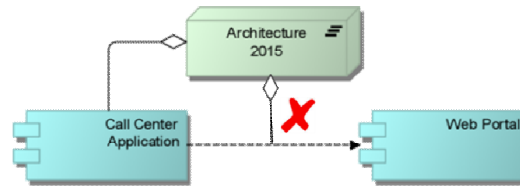


Figure 39: Scenario 2 – Target and source components

3. Source component is not valid but target component is valid to a certain plateau. Here, aggregating a relationship is not possible and not allowed because the plateau only knows the target component. Even though the relationship itself knows the source and target components by storing this information on its attributes, the plateau does not know the source component. This happens because only the target component is valid and aggregated to the plateau. If aggregating such condition is allowed, then the logic of the relationship from the perspective of the plateau would not be correct. Therefore, it is considered not possible and not allowed to aggregate a relationship where only target component is valid to the plateau.

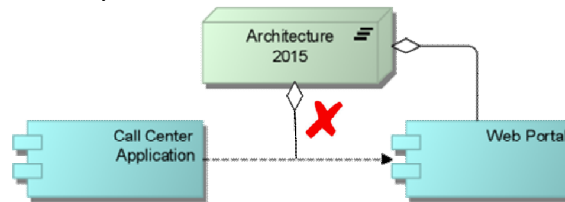


Figure 40: Scenario 3 – Target and source components

4. Both source and target components are not valid to a certain plateau. The last scenario shows that none of the source and target components are aggregated to the plateau. In this case, the relationship between the components that are both not part of the plateau could not be aggregated to the plateau. It would not make sense to state that a relationship between two components belongs to a plateau where neither the source component nor the target component is valid and aggregated to the plateau. Therefore, it is considered not possible and not allowed to aggregate a relationship within such context.

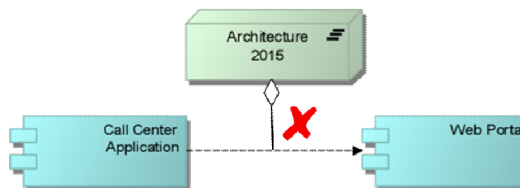


Figure 41: Scenario 4 – Target and source components

The **second concern** of aggregation extension to plateau concept is about the aggregation definition itself in ArchiMate metamodel. Aggregation relationship is categorized as structural relationship that models the structural coherence of concepts of the same or different types. Aggregation relationship is used to indicate that an object groups a number of other objects. Aggregation relationship is always possible between two instances of the same objects. The main concern in the ArchiMate metamodel definition of aggregation relationship is that it only defines aggregation among concepts.

In ArchiMate, there is clear distinction between core concepts of the language and the set of core relationships of the language. This proposed aggregation extension requires that aggregation relationship in a plateau enables aggregation not only about concepts but also about relationships. Therefore, it is suggested in the new extension to accommodate that aggregation is also possible between instances of the concepts and also relationships among them.

2. “Grouping” relationship to represent plateau concept.

“Grouping” relationship could be utilized in extending the definition of plateau concept. This alternative solution would impact the ArchiMate specification for plateau definition as would be described in implications sub-section. Grouping relationship share common characteristics to support plateau concept definition. In ArchiMate, grouping relationship indicates that objects belong together based on some common characteristics. Plateau concept defines a relatively stable state of the architecture that exists during a limited period of time. Here, the common attribute or characteristic of the objects, or concepts, is the validity within a limited period of time.

Several objects can belong together to form a plateau based on the date validity attribute. A plateau is considered as a snapshot of the architecture for a relatively stable period of time. Thus, all objects within the architecture that are valid at some points in time are considered part of the plateau, as long as their start date and end date match with each other. For example, Architecture 2015 is the plateau that is valid from year 2015 onwards. This result in a situation where all architectural elements or concepts or objects that are valid in year 2015 onwards could also be grouped together as part of the Architecture 2015 plateau.

Grouping relationship could group several model objects that are of the same type or of different types. The collection of the model objects can also be from different architecture domains (across architecture layers). This characteristic aligns with the required condition of a plateau concept where architectural elements (concepts/objects) of the same or different types could form together to represent an enterprise architecture state (plateau). The architectural elements are not only from various types of elements but also from various or multiple architectural domains (business layer, application/information layer and infrastructure/technology layer).

Moreover, grouping relationship allows the model objects to belong to multiple and overlapping groups. This characteristic of grouping relationship also accommodates the plateau concepts requirements where an architectural element could be part of more than one plateaus. For example, a General CRM system application is a part of both baseline architecture and target architecture. This is possible, for instance, because the transformation roadmap does not require the General CRM application to be modified or removed in order for the enterprise to realize its target architecture. Up to this point, grouping relationship could be seen as a good way to depict the relationship between the plateau and all of the core components and relationships that belong to it.

Implications

Several implications or adjustments need to be made explicit to the existing grouping relationship definition in ArchiMate metamodel when this solution is to be in place. The first implication is to state explicitly that the grouping in this case is made specifically in the context of plateau definition. Like grouping in general, grouping relationship could be used for any kind of grouping purpose, which is to show that several components being grouped share something in common.

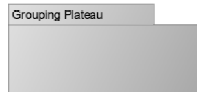


Figure 42: Grouping relationship for plateau definition

The second implication is to define a relationship between grouping and plateau concept. In the current ArchiMate metamodel definition, a grouping relationship could not be assigned a relationship to any other components because there is no definition or specification for that. Moreover, a grouping relationship does not form an “overall” object of which the grouped objects form a part. When all of the related and relevant objects and relationships have been grouped together graphically as a plateau, the next step is to have a relationship from the group to the plateau concept.

An “assignment” relationship from the grouping plateau to the plateau concept could be used to indicate that all of the objects and relationships grouped together within the group belong to the plateau being assigned. The figure below shows the same meaning. It shows how the grouping relationship could be used to express the plateau concept definition.

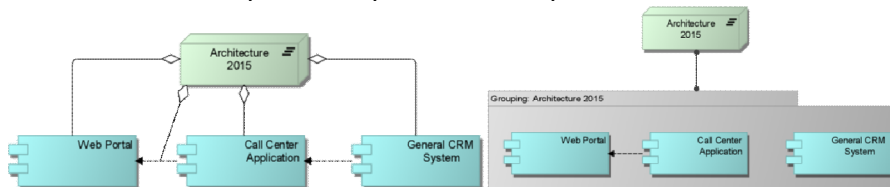


Figure 43: Plateau - grouping relationship

By referring to the figure above, it could be inferred that everything that is inside the grouping, is also part of the plateau concept. For example, all three application components are inside the grouping and thus, also part of the plateau. The flow relationship from call center application to web portal is drawn and included in the grouping. Thus, this flow relationship is also part of the plateau. Notice that the flow relationship from general CRM system to call center application is not included in the grouping. Therefore, this flow relationship is not part of the plateau. Finally, both images (the left and the right) depict the same meaning.

However, several important notes or challenges need to be considered when implementing the grouping relationship to define plateau concept. Grouping relationship does not result in an “overall” object that is formed by the objects that share common characteristics. In figure above, “Grouping: Architecture 2015” is not an object. The object form is only to represent graphically that the objects and relationships placed within it are grouped based on certain commonality. The plateau, in this case, is the “overall” object of which the grouped objects form a part by

having the aggregation relationship.

In ArchiMate modeling language, grouping relationship does not have formal semantics. It is purely used to show graphically that model elements have something in common. This limitation would impact future analysis, for example in performing gap analysis between two plateaus. When there is no formal semantics between the group and the grouped objects, then it would be even more difficult to analyze the gaps between plateaus. This is because the tracing functionalities would not be (much) supported while identifying the differences between two plateaus.

Since there is no current formal semantic for grouping relationship, it would also be difficult to filter or clearly express which relationships between components are parts of the group. For example the relationship between web portal and call center application are drawn indirectly to the plateau. This indirect expression of relationship would also impact the future analysis needed to be performed with regards to plateaus, for example gaps analysis or roadmap visualization purpose.

Having grouping relationship to address this conceptual problem requires more effort and modifications when compared to the core fundamental problem that needs to be solved. The existing definition of plateau concept in ArchiMate metamodel does not involve grouping relationship as intermediary between the plateau and the objects/concepts and the relationships. Moreover, this grouping relationship solution completely changes how the plateau concept has been defined in the current ArchiMate specification. Implementing this solution means using grouping relationship to represent that several objects and relationships are part of the plateau. Therefore, the “alignment” criteria in proposing a solution becomes crucial since this specific solution would not apply the already defined specification.

3. Introduce new type of aggregation relationship.

Another alternative solution is to introduce a new type of aggregation relationship. This introduction would impact the ArchiMate metamodel because new type of relationship is introduced. The idea of the new type of aggregation relationship is to allow the relationship attributes of the component being aggregated to a plateau to be also part of (aggregated to) the plateau.

This new type of aggregation relationship is described as follow. The fundamental of ArchiMate modeling language is very much similar to the object oriented modeling language. In object-oriented modeling language, an object would have several attributes which define and explain about the characteristics of that particular object. In ArchiMate modeling language, this feature is also supported even though it is not explicitly shown in the model canvas. The properties or attributes of an architectural element could be defined and specified in order to provide more comprehensive information about that particular element. For example, an application component (an active architectural element in application/information domain) might have its name, application status, application usage, application total cost and any other relevant information stored in its properties.

The information about the element's relations to other architectural elements is also stored in the properties, or called attributes in object-oriented modeling language. For example in BiZZdesign Architect tool, the figure below shows how the properties or attributes of the architectural elements are stored.

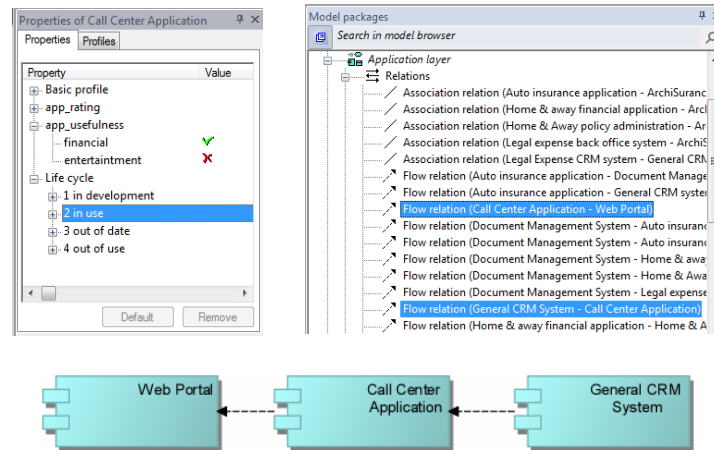


Figure 44: Architectural element's properties and relationship attributes

By referring to the previous explanation, the newly proposed type of aggregation relationship could be made possible. In this thesis, the newly proposed type of aggregation relationship is referred as “plateau aggregation relationship”. Plateau aggregation relationship is a specialization of aggregation relationship that is specific only in the context of plateau definition. Plateau aggregation relationship indicates that a plateau aggregates a number of other concepts including the relationships attributes hold by the concepts.

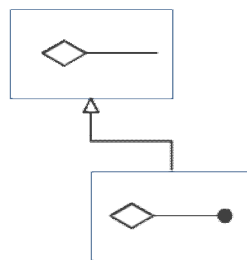


Figure 45 : Plateau aggregation relationship: specialization and notation

The application of the plateau aggregation relationship is described as follow. By using the same example used in the first solution, the plateau definition is depicted in figure below. As depicted in figure below, this new aggregation relationship states that web portal and call center application belong to Architecture 2015 plateau. The dot symbol, as the additional thing to show the plateau aggregation relationship, at the other edge of the aggregation relationship states that, or could be inferred as, the relationship attribute of an object could be aggregated to the plateau as well.

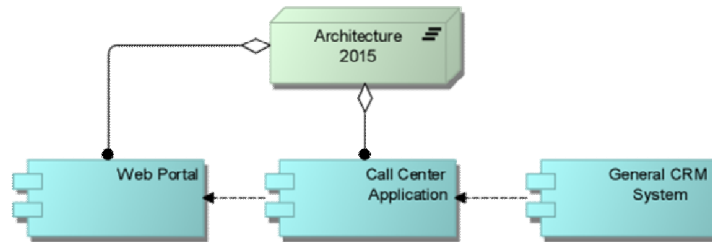


Figure 46: Proposed type of aggregation relationship

In this case, since web portal application and call center application are both part of the plateau, then the flow relationship between these two applications are also part of the plateau. Thus, the additional dot in the plateau aggregation relationship makes it possible to also include the relationship between components to be explicitly stated and defined as part of the plateau concept as well.

Take “call center application” as an example. By having the new plateau aggregation relationship from the application component to the architecture 2015 plateau, it expresses two things. The first is to express that call center application is aggregated to the plateau. The second is to express that the aggregation might also include any relationships owned by the call center application to be aggregated to the plateau as well. The metamodel is now then able to include the relationship between core elements to be also part of the plateau. This solution proposition makes an explicit representation of plateau aggregation concept.

Implications

There are several concerns that need to be addressed while implementing and applying the newly proposed plateau aggregation relationship. The first concern is that introducing a new type of relationship into the ArchiMate metamodel means making the modeling language even more complex. The existing number of relationship types in ArchiMate is quite high and from those various types of relationships, only several relationship types are used intensively and extensively by the users.

The second concern is that the additional type of aggregation relationship has a very specific context. The application of the plateau aggregation relationship is only limited to the plateau definition. Therefore, while considering the effort needed to introduce the new relationship type and compared it to its implementation context, the higher complexity and modification is somehow not worth trying.

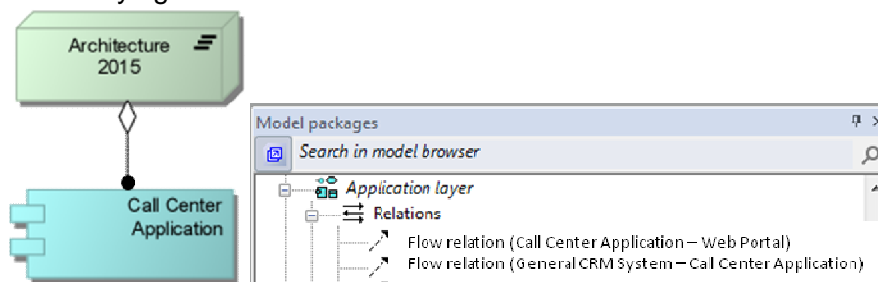


Figure 47: Call center application plateau aggregation relationship

The third concern is that there is no clear and explicit statement or assignment about which of the relationships attributes owned by the object is/are aggregated to the plateau concept. Although it makes aggregating a relationship to a plateau possible but it does not state which relationships are being aggregated. Referring to the previous example as shown in figure above, the call center application has two relationships: (1) is the data flow relationship to web portal application and (2) is the data flow relationship from general CRM system.

Knowing that only web portal application is also aggregated to be part of the Architecture 2015 plateau and the general CRM system is no longer valid in the year 2015, it could be concluded that only relationship (1) of call center application is allowed to be aggregated to the plateau. Relationship (2) could not be aggregated to the plateau because the general CRM system is not aggregated to the plateau. Illogical relationship would be derived if relationship (2) is also aggregated to be part of the plateau because then, from the perspective of Architecture 2015, the relationship (2) does not have source component of the relationship.

Solutions comparison and selection

The previous part of this section has discussed comprehensively the three alternative solutions to address the aggregating to a relation problem. This part of the section is now comparing the three alternative solutions in order to come up with the best fit proposed solution of the thesis. There are three criteria that will be used as the basis for assessing and comparing the alternative solution: *does it solve the core problem?*, *does it require many modifications?*, and *is it feasible to be applied in the existing enterprise architecture tool?*

Does it solve the core problem? The first criterion is assessing whether the alternative solution addresses the core problem or not. As previously mentioned, the core problem of the aggregating to a relation problem is the lack of aggregation possibility to support that a relationship between concepts or objects or components could also be aggregated to plateau concept. The first alternative solution, *adding relations aggregation to plateau concept*, addresses the core problem comprehensively because it allows the aggregation relationship to be directly link the relationship between components to the plateau. This additional aggregation relationship fulfills the basic need of the problem. The second alternative solution, *grouping relationship to represent plateau concept*, does not directly add aggregation relationship from the relationship between components to the plateau. It serves the purpose by having grouping relationship as intermediary (indirectly) to link the relationship to the plateau concept. Thus, the second alternative solution does not fully and comprehensively address the core problem. The third alternative solution, *introduce new type of aggregation relationship*, somehow solves the core problem under some conditions. The new type of aggregation is called plateau aggregation relationship, which is a specialization of an aggregation relationship. Even though it enables the connection of a relationship between components to the plateau, it does not specifically and explicitly state which relationship attributes of components are aggregated to a certain plateau. Thus, this alternative solution addresses the core problem semi comprehensively. Therefore, when analyzing whether or not the solutions address the core problem, the first alternative solution is the most satisfying alternative, followed by the third alternative solution, while the second alternative solution is the least satisfying one.

Does it require many modifications? The second criterion is assessing whether the alternative solution requires many modifications to the ArchiMate modeling language in order to be implemented, including the implications. The first alternative solution, *adding relations aggregation to plateau concept*, does not require many modifications to the existing ArchiMate metamodel with regards to the implementation and migration extension. The only adjustment needed is the additional part of having an aggregation relationship from the relationship between components to the plateau concept. There is no new type of relationship being introduced. However, the concept aggregating a relationship to an object is never introduced or implemented before. Aggregation relationship has always been used to express that some objects might gather themselves to form a bigger unification object. Thus, aggregation relationship is always dealing with objects with other objects. The second alternative solution, *grouping relationship to represent plateau concept*, completely changes the way the plateau concept is defined. It also requires an assignment relationship from a grouping to a plateau concept. The plateau-grouping relationship only applied to the condition where grouping relationship will be dedicated to represent plateau and thus, is only applied to the plateau definition context. The third alternative solution, *introduce new type of aggregation relationship*, requires quite big modification to the ArchiMate metamodel. It is due to the fact that this alternative solution introduces new type of aggregation relationship. More analysis needs to be in place in assessing the implications of introducing the new type of aggregation relationship than what have been performed in this thesis. The introduction of new aggregation relationship type complicates the whole list of ArchiMate relationships. Thus, the third alternative solution receives very low value when it comes to the second assessment criterion. Therefore, when analyzing whether or not the solutions require huge modifications to the existing ArchiMate metamodel definition, the first alternative solution is the most satisfying alternative, followed by the second alternative solution, while the third alternative solution is the least satisfying one.

Is it feasible to be applied in the existing enterprise architecture tool? The third criterion is assessing whether the alternative solution is feasible to be practically implemented in the existing enterprise architecture tool. The complexity and efforts need to be invested are also being evaluated. The first alternative solution, *adding relations aggregation to plateau concept*, is very feasible to be implemented in the existing enterprise architecture tool. In fact, BiZZdesign Architect is able to perform this functionality. The second alternative solution, *grouping relationship to represent plateau concept*, is considered to be feasible to be implemented in the existing enterprise architecture tool because it requires less modifications or introduces less changes to the existing grouping relationship. What should be noted in mind is that for the new grouping relationship to perform future analysis and visualization with regards to plateau utilization, for example gaps analysis and roadmap visualization, more efforts are needed. The second alternative solution provides indirect connection from the relationship between components to the plateau concept. Thus, when performing analysis which involves this indirect relationship, the logic of the script would be also more complex. The third alternative solution, *introduce new type of aggregation relationship*, requires the highest effort of modification among the three alternative solutions. This is because it introduces something completely new and never been implemented before. The efforts needed to consider the analysis and visualization

which are related to plateau concepts will also be huge. Thus, the third alternative solution receives very low value when it comes to the third assessment criterion. Therefore, when analyzing whether or not the solutions are feasible to be implemented in the existing enterprise architecture tool, the first alternative solution is the most satisfying alternative, followed by the second alternative solution, while the third alternative solution is the least satisfying one.

Table 5: Summary of alternative solutions assessment

Criteria	Weight	Solutions					
		Alternative #1		Alternative #2		Alternative #3	
<i>Does it solve the core problem?</i>	50	10	500	6	300	7	350
<i>Does it require many modifications?</i>	25	8	200	7	175	5	125
<i>Is it feasible to be applied in the existing enterprise architecture tool?</i>	25	10	250	6	150	5	125
Total Points	100	950		625		600	
		95		62.5		60	

The above table summarizes the criteria evaluation and assessment for all of the three alternative solutions. According to the three criteria assessment, it is clearly shown that the thesis selects the first alternative solution: *adding relations aggregation to plateau concept*. It should be noted that this selection does not guarantee that the chosen solution would completely solve the problem in practice. The selection is performed in comparison with the other two alternative solutions.

4.2.2 Practical Solutions

The following three solutions are dealing with the practicality or pragmatic usage of the enterprise architecture tool with regards to the transformation process purpose. These solutions aim to improve BiZZdesign Architect functionality in helping and assisting the users. Thus, these solutions are mostly related and specifically dedicated to BiZZdesign Architect.

S2. Plateau duplication functionality

This practical solution takes into account the previous discussion about the necessity of having “plateau” as a concept in ArchiMate modeling language. Rather, “architecture” as a concept should be defined and “plateau” would serve as a state of the “architecture” concept. Since the plateau concept is used to represent the states of architecture, it could be concluded that plateau is a snapshot of architecture. The period of the architecture snapshot is, in this case, not a short period. As the definition of the plateau concept states, plateau represents a relatively stable state during a limited period of time. The states of the architecture that are covered by the plateau are baseline, transition and target.

Commonly, different states are built upon each other and there is a need for an existing state, baseline architecture for example, to build the foundation for the definition of a new state, target architecture for example. The target architecture could be defined based on the already

described baseline architecture with several changes or modifications. Therefore, it is recommended that the functionality of plateau duplication should be in place. This functionality or mechanism should allow the process of duplication of a plateau (an enterprise architecture) which is then treated as a new plateau (a new state of enterprise architecture) to represent a new state of the architecture. Subsequent changes to the newly created plateau should be performed independently from the original plateau. By having this, it will promote the practicality of plateau creation since the enterprise architect could duplicate an existing current state and then independently apply changes that lead to a final representation of a target state.

Some concerns need to be considered when implementing the solution in the tool. BiZZdesign Architect tool has two types of copying or duplication treatments which result differently in terms of architectural components list. The first type of duplication is when the user performs the duplication process on the view screen of the architecture. This action will result in duplicated components graphically but not in the structure list of components. The duplicated component will only have one entity on the tree structure even though there is more than one graphic representation of the components. The second type of duplication is when the user performs the duplication process on the tree structure section of the tool. This action will result in duplicated components on the tree structure as new entity and thus, will have new graphical representation.

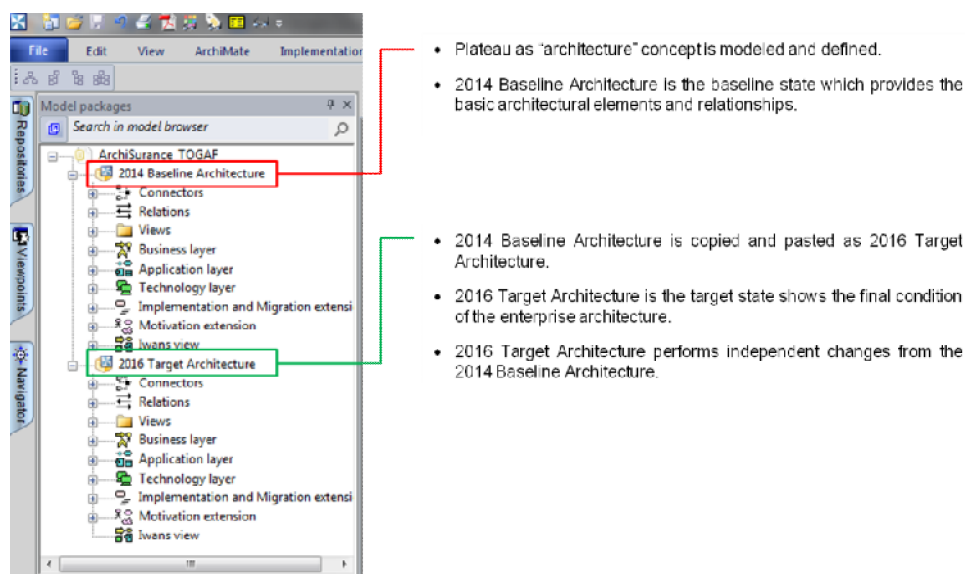


Figure 48: Illustration of plateau duplication functionality

This solution considers the second type of duplication action where the newly duplicated plateau is treated as a new entity along with all of the core components and relationships it has. Since they are considered as the new entities, then independent modifications to depict the changes in the new plateau should be made possible. Therefore, it requires careful execution or implementation while developing this functionality. The tool must be very careful when deciding which copy-paste action would lead to creating new entity in the architecture tree structure, or architecture repository.

Another concern while replacing “plateau” concept with “architecture” concept relates to future analysis or visualization-oriented activities. When introducing the concept of having “architecture” as a model in architecture tree, it should be noted that it will impact the plateau operation activities. In ArchiMate, a plateau could trigger another plateau by executing work packages or transformation projects. This work packages would deliver required changes or modifications to the architecture, for example certain applications need to be removed; a new application needs to be introduced to replace the functionality of several old applications, etc.

By having “architecture” as a model itself, and not as a concept within a model, then it would be difficult for the BiZZdesign Architect tool to define the trigger to the plateau. Moreover, roadmap plan would even be more difficult to be made because the plateau is not part of the model itself. This is because all of the analysis performed within the tool always deal with concepts that are defined within a model. It would be interesting to learn whether or not the tool allows analysis across multiple models. If such thing is possible, then implementing this solution would also be made feasible.

This practical solution requires huge effort in knowing the existing functionality and capability of the BiZZdesign Architect tool in order to evaluate if it is something feasible and worth doing. Due to the limited time and knowledge of the author about the tool itself, a comprehensive assessment of this solution implementation could not be made. However, implementing this would help the users of BiZZdesign Architect tool in creating new states of their architecture enterprise in a more pragmatic and efficient way.

S3. Definition view of a plateau

As previously described in section 3.1.3, there are two approaches in plateau definition that could be performed in BiZZdesign Architect: (1) through properties or profiling and (2) through aggregation relationship. Plateau definition through profiling shows that an architectural component belongs to a certain plateau by having the properties of the respective component updated. The field about plateau is then to be updated with the respective plateau. This is done before the BiZZdesign Architect was upgraded to accommodate the implementation and migration extension concepts.

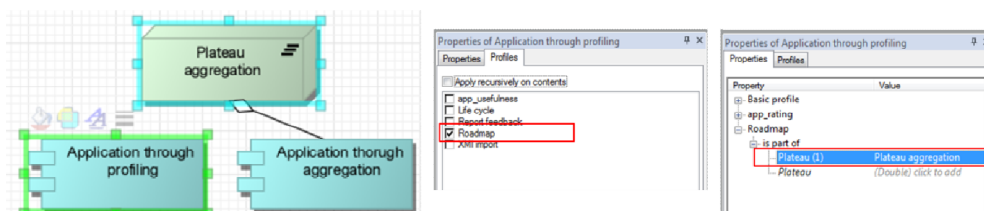


Figure 49: Plateau definition through profiling

Plateau definition through plateau aggregation is another way of setting an architectural component to be part of a certain plateau by directly connecting the desired architecture’s component to desired plateau (drag and drop aggregation relation). This approach was made possible after the BiZZdesign Architect was upgraded to accommodate the implementation and

migration extension concepts.

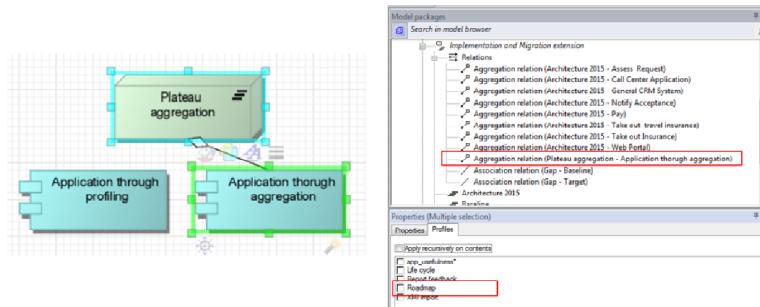


Figure 50: Plateau definition through aggregation relationship

In current version of BiZZdesign Architect, both ways are still possible to be performed by the users. The figure below shows the graphical representation of a plateau and the two components that belong to that particular plateau. By looking at the figure, only one application is explicitly shown as part of the plateau. This is because there is clear aggregation relationship line from that application to the plateau. This is done by using the (2) approach. Meanwhile, the other application, by only looking at the graphical representation, could not be seen as part of the plateau because there is no clear line that shows the aggregation condition. This situation is the result of the (1) way of plateau definition. This is because the assignment process that states that an application is part of the plateau is done at the back of the BiZZdesign Architect. Thus, this relation could not be seen directly by the users.

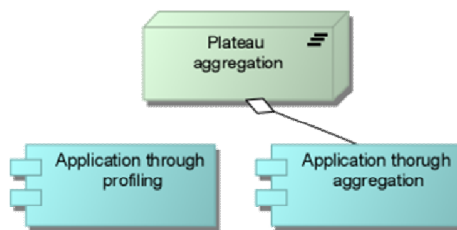


Figure 51: Two ways plateau definition in BiZZdesign Architect

Even though the users are provided with two possible ways to define plateau, the process of assigning the components to the plateau requires big effort not because of the complexity of the action but because of the repetitiousness of the action. For each of the components and relationships that need to be part of the plateau, individual assigning action must be performed. Imagine if an enterprise has so many architectural components or elements that are part of a certain plateau, then the plateau definition process is really troublesome.

It would be handy, if BiZZdesign Architect could provide plateau aggregation with multiple components at one time. This could be done theoretically in two approaches. The first approach is by selecting multiple relevant components that need to be aggregated to the plateau. After the components have been selected, there should be an additional functionality that allows the users to define aggregation relationship to a particular plateau. Right-click functionality from the "smart connector" could be added to accommodate this multiple aggregation action. After the users have aggregated the components to the desired plateau, then on the architecture tree, the list of aggregation relationship to the plateau must be automatically created. In order to maintain

its model simplicity to the users, the actual lines of aggregation relationship could be omitted from the model display. The figure below shows how this could be implemented in BiZZdesign Architect.

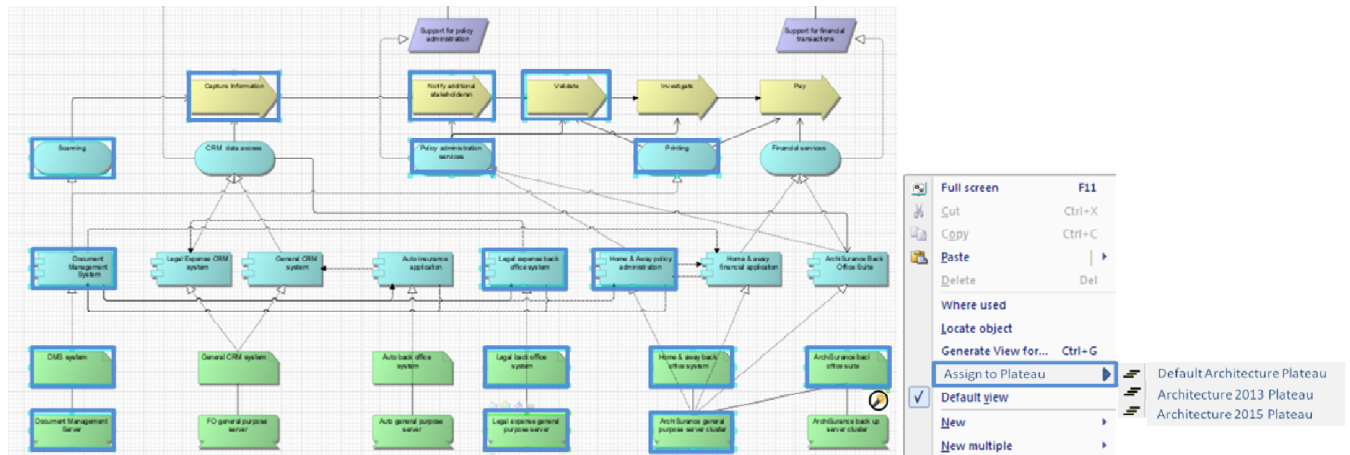


Figure 52: Multiple components aggregation to plateau

Another approach to this multiple components aggregation to the plateau is by introducing a dedicated view for plateau definition. This view definition is of “*total view*” type in BiZZdesign Architect tool. The *total view* type is considered suitable for plateau definition because it could allow any types of architectural elements from multiple architecture domains, including the motivation extension and implementation and migration extension concepts.

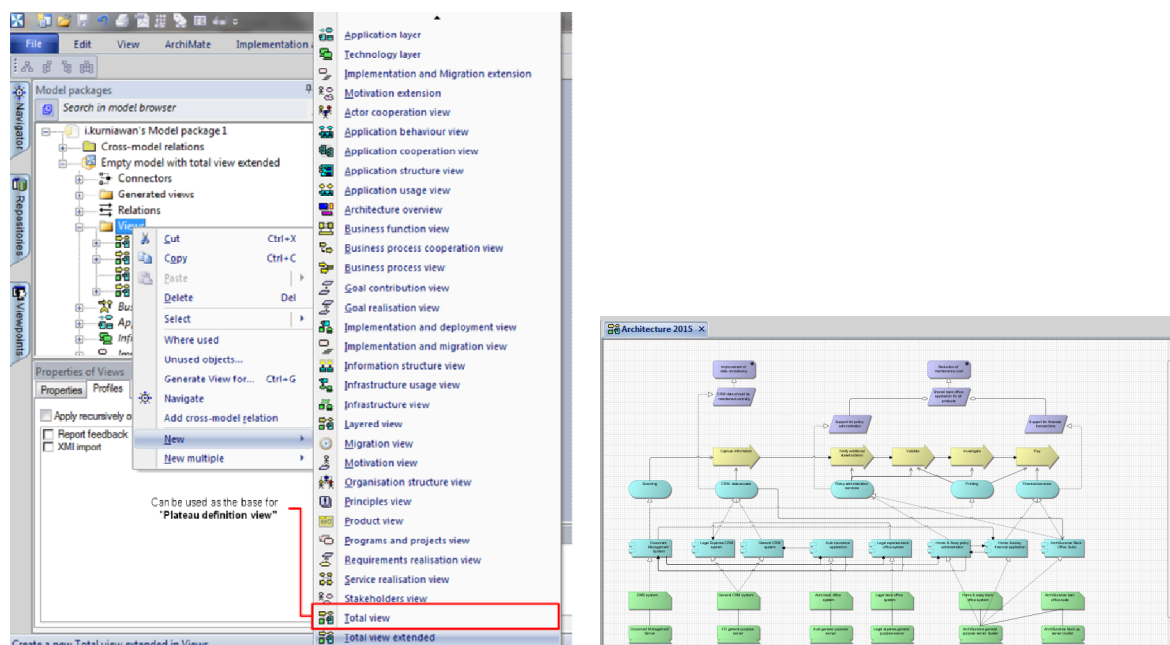


Figure 53: Total view to define plateau aggregation

When a total view is created, a blank canvas model will be available. Then, the users can put any related architectural elements that belong to the plateau's definition view. When a

component is placed in the model, then automatically an aggregation relationship between that component and the plateau concept is also created. Likewise, when a component is taken out or removed from the model, then the aggregation relationship between the component and the plateau is also deleted. Thus, it could be inferred that the plateau definition model acts as a representative of aggregation relationship when defining a plateau. The plateau definition model will be linked to the plateau concept in the model.

In the existing support of Architect, there is no definition view to see which core architectural elements belong to a specific plateau. Currently, the plateau concept is represented in an object form as part of the implementation and migration extension concepts. The aggregation relation between the core elements and the plateau is stored in the core elements' relations list. However, the support of viewing a specific plateau, in other words a specific state of architecture, is limited and minimum. Direct and easy viewing functionality will be beneficial for the users to see particular states of their enterprise architecture.

The plateau definition view, with regards to S1 above, should then be displaying the architecture view which consists of core architectural elements along with the relevant relationship among them. When S1 is not currently supported, the definition view of the plateau would then only consist of the core architectural elements without the relationship among them. This would make the view less meaningful as it does not show e.g., what the relationship is between two application within the plateau.

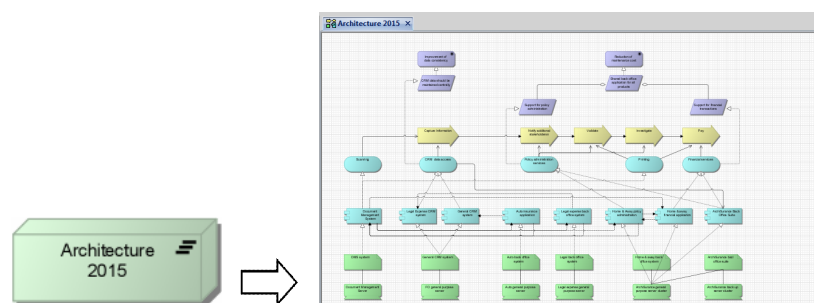


Figure 54: Plateau concept and its definition view

As shown in the above figure, when the functionality of “plateau definition view”, which utilizes the use of “total view” in the existing BiZZdesign Architect functionality, then showing or displaying the architectural components and their respective relationships within the plateau (aggregated to the plateau) is made possible. The plateau definition view could be similar to the “layered viewpoint”. It pictures several layers and aspect of enterprise architecture in one diagram. The main goal of the layered viewpoint is to provide overview in one diagram. This could also be used as support for impact of change analysis or for extending the service portfolio.

S4. Gap concept view

This solution is less related to aggregating a relation problem. It is more related to the concern of components relationship in different plateaus. As listed in section 3.2, there is a difficulty to

show the relationship between applications in different plateaus. This problem arises especially when there is one core element removed and a new core element is added.

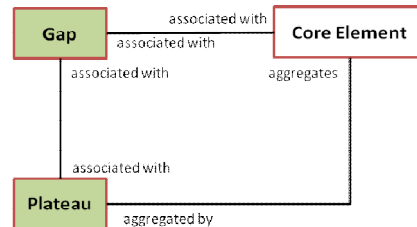


Figure 55: Gap, Plateau and Core element interrelation, derived from The Open Group (2012)

The current support of the tool is lacking in showing the clarity whether the newly added core element replaces the removed core element. The new element could be possibly having no relation at all with the removed element. Therefore, it is hard to directly conclude the relation between the removed and added core elements. This limitation could be addressed by utilizing the gap concept that has already been defined in ArchiMate modeling language. As can be seen from Figure 30 in section 4.1.1 previously, the concept of gap, plateau and core element are connected. This interrelation among them enables the initiative to further extend the functionality of gap analysis process.

Each of the core elements of the enterprise architecture will be aggregated to plateau(s) in order to show their validity over time period of certain plateaus. There are at least four (4) transformation patterns that describe what happens to the gap components in between the plateaus' definition. The four transformation patterns or conditions are: (1) removed, (2) added, (3) modified, and (4) replaced.

- (1) **Removed.** In removed transformation pattern, the architectural components are valid only in the previous plateau and are no longer valid in the period of the new existing plateau. This means that the components are being removed and do not exist in the running enterprise architecture's state. When the removed components are replaced by any other components, either the already existing components, modified components or newly added components, then they are classified as "replaced" (see point 4 below, as the being replaced components). The architectural components are just simply being removed, and are not valid in and part of the new enterprise architecture state.
- (2) **Added.** In added transformation pattern, the architectural components are valid in the new existing plateau. These components have not been defined in the previous plateau and have only been defined in the new plateau. When the added components are introduced to replace any other removed components, then they are classified as "replaced" (see point 4 below, as the replacing components). They could be introduced as completely new components not replacing any other components. The architectural components are just simply being introduced, and thus added, in the new enterprise architecture state.
- (3) **Modified.** In modified transformation pattern, the architectural components, or core concepts, are valid both in the previous plateau and the new existing plateau. This means that these components are part of both enterprise architecture's states. What they

experience in between the plateaus is the modifications or updates within the components. The architectural components, however, are still considered the same and are only modified. Therefore, the modified transformation pattern does not result in newly added components.

- (4) Replaced. In replaced transformation pattern, there are architectural components, or core concepts, that are being removed ("being replaced") and there are other architectural components replacing the removed components ("replacing"). The architectural components that are replacing the removed components could be newly added components, modified components or unchanged components.

Therefore, it could be derived that gap concept could be further classified into four types of gaps relationship, or in this thesis is called transformation pattern. The classification of these types of transformation patterns could be seen in the figure below.

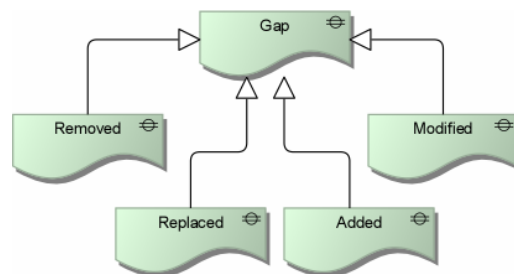


Figure 56: Gap concept classification

The information about the relations between the removed, added, modified and replaced core elements could be defined in gap concept. The figure below illustrates the gap analysis view between two plateaus: baseline and target plateau. Application A, B and C will be removed and no longer valid in the target plateau. Application X and Y are introduced only in the target plateau and thus, being added. Application C is valid in both baseline and target plateaus. Suppose the information about the relations between these gaps components are known as follows. Application B and D will be replaced by Application X. Application C experiences modifications or updates when transforming from baseline plateau to the target plateau. Application A is simply being removed from the (baseline) architecture without being replaced by other components. Application Y is simply being added to the (target) architecture without replacing any other removed components.

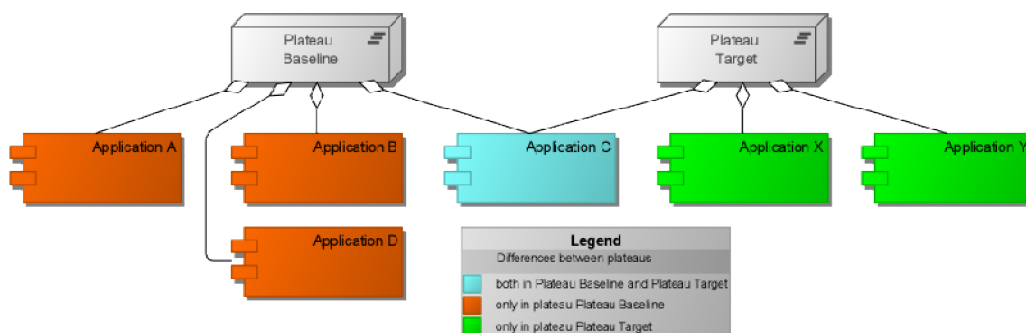


Figure 57: Illustration of gap analysis of plateaus transformation

Based on the above gap analysis illustration and context, the following gap concepts' classification could be drawn. The figure below shows that Plateau Baseline triggers the existence of Plateau Target, which in ideal and complete case the transformation would be realized by implementing several transformation projects, or work packages. Since this practical solution is only dealing with making a clear relationship among the gap components, then the implementation projects or work packages are excluded.

This solution is feasible to be implemented in the current version of BiZZdesign Architect because no additional concept is introduced. The solution utilizes the already available functionality/function to serve more various purpose of view.

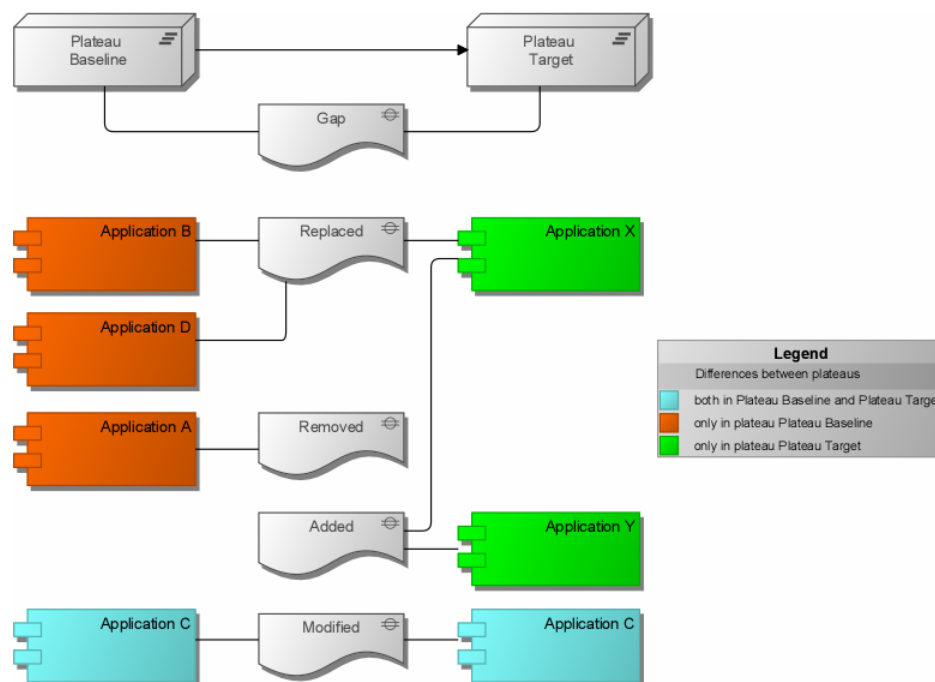


Figure 58: Illustration of gap concept classifications

4.3 Solution to “Consolidate Gaps, Solutions and Dependencies Matrix” Problem

As previously described, the Consolidated Gaps, Solutions and Dependencies Matrix is used as a planning tool when creating the work packages. The aim is to review, consolidate, and integrate the gap analysis results from the Business, Information Systems, and Technology Architecture, which are performed in phase B, C and D respectively. The guideline on how to come up with the matrix is still lacking. Beside coming up with the list of work packages to close the gaps, another important goal of the matrix is to consolidate the gaps. Valuating the gaps could be the result of consolidation process which later could be used as the foundation to make gaps portfolio in order to prioritize which gaps are needed to be closed first in order to reach the target architecture.

Since the matrix development is part of phase E, this study assumes several inputs and information are readily available as the outcomes of the previous phases. Particularly, phase B, C, and D are considered the core of the enterprise architecture generation and thus, highly important. Phase B deals with the business domain of the architecture. Phase C deals with application domain of the architecture, including the data domain. Application domain is sometime referred to as information systems architecture. Phase D deals with the technology domain of the architecture. Therefore, the following inputs are assumed available in order to execute the proposed method which will be described afterwards.

1. Gaps analysis results of business architecture
2. Gaps analysis results of information system (application) architecture
3. Gaps analysis results of technology architecture
4. The following viewpoints:
 - a. Goal and Requirements realization viewpoints. To see how the requirements, resulted from goal, are realized by the core elements, such as business actors, business services, business process, application services, application components, etc.
 - b. Business process co-operation viewpoint: To see the relationships of one or more business processes with each other and/or with their environment. It is also important to see the realization of services by business process
 - c. Service realization viewpoint. To see how one or more business services are realized by the underlying processes, and sometimes by application components.
 - d. Application usage viewpoint. To see how applications are used to support one or more business processes, and how they are used by other applications.
 - e. Infrastructure usage viewpoint. To see how applications are supported by the software and hardware infrastructure. The infrastructure services are delivered by the devices; system software and networks are provided to the applications.
 - f. Layered viewpoint. To see the overall picture of several layers (domains) and aspects of an enterprise architecture in one diagram.

4.3.1 Gaps Portfolio Valuation

The proposed process of consolidating the gaps in order to come up with gaps portfolio valuation is shown in Table 6. The proposed method consists of seven main steps which takes insights from previous research, especially in the area of IT portfolio valuation.

Table 6: Proposed steps of gaps portfolio valuation

Step 1	Collect gaps' components
Step 2	Group interrelated gaps' components
Step 3	Filter out the model
Step 4	Calculate the importance of the components' group to the organization
Step 5	Calculate the effectiveness of the components' group to the organization
Step 6	Calculate the interdependency level of the components' group
Step 7	Prioritize gaps' components groups

Step 1: Collect gaps' components

The first step of the gaps portfolio valuation is to collect or list down all of the gaps components between two plateaus. The motivation of this step is simply to have a repository of the architectural components that make up the gaps between two plateaus, and thus, they are called gaps components.

Gaps' components can be collected from the gap analysis results of business architecture, application architecture and technology application. In this study, we mainly limit the gap analysis results into the behavior elements which perform the operation of the organization's business. Gaps' components resulted from gap analysis of business, application and technology architectures are limited to business process, application components and nodes.

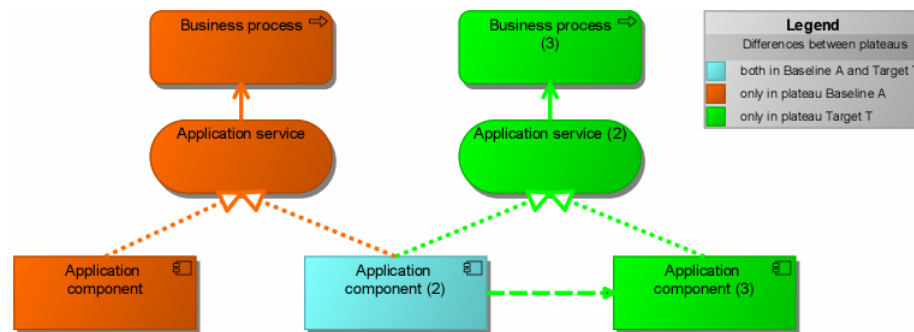


Figure 59: Example of gaps analysis view

```

for all objects in model objects
  if object has plateau aggregation relationship
    if object is aggregated to plateau baseline
      add object to 'plateau baseline component'
    if object is aggregated to plateau target
      add object to 'plateau target component'
    if object is aggregated to plateau baseline AND plateau target
      add object to 'plateau all component'

for all objects in 'plateau all component'
  if object is being modified
    add object to 'modified component'

gaps components is
  ('plateau baseline component' + 'plateau target component') -
  + 'modified component'

```

Gaps' components can be categorized into three major categories based on the existence or validity: modified, removed and added. Modified components are the components that exist in baseline architecture but need to be modified and are still valid in the target architecture. Removed components are the components that are no longer valid in the target architecture. Added components are the newly added component in the target architecture. The new components could be the result of unification of several components, the replacement of the removed components or completely new

components. This gap categorization would be possible when the practical solution, S4, in section 4.2.2 has been implemented.

Step 2: Group interrelated gaps' components

The second step is to group the collected gaps components based on the (change) goals of the organization. The change goals are the goals driving the organization to transform from its baseline, or as-is, architecture into the target, or future, architecture. The motivation of this step is to have an overall overview of the collection of gaps components that would contribute to the realization of the goals. This step is important in order to align the business strategy, which is mostly stated as the organization's goals, to the IT strategy. IT strategy could be seen from the work packages (IT projects) being selected to be implemented to reach the target architecture. Reaching the target architecture means closing the identified gaps previously identified. Thus, having an overview of the relations between the gaps components to the organization's goals is crucial.

The gaps components of the architecture will have, either directly or indirectly, the correlation to the main goals. If the relation between the gaps components to the main goal is explicitly stated, then the algorithm to trace the relation is straight forward. However, if there is no direct relation from the gaps components to the main goals, then an algorithm for a deeper analysis is needed. A business goal could be further divided into sub-goals and sub-sub-goals. Goals and sub-goals must be aligned with certain principles that are being held within an organization. Therefore, in order to reach some goals in accordance to the principles, some requirements could be derived. In most cases, requirements are realized by the components of the architecture. By tracing the path from main goals to sub-goals, and to principles and requirements, and to components; a relation from gaps components to the main goals could be drawn.

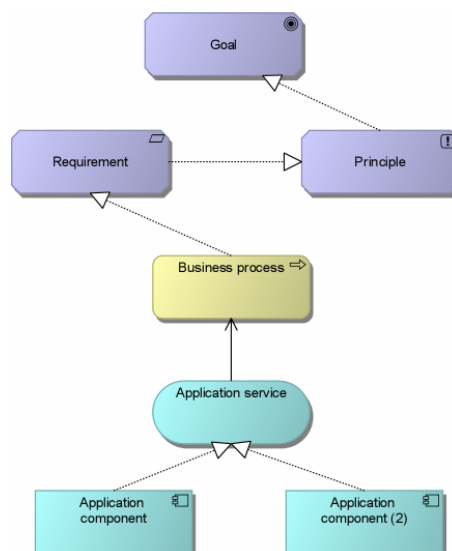


Figure 60: Goals – components relationship

Grouping the gap components with respect to the (change) goals or the organization is one criterion that would be considered. Another criterion in grouping the gap components is the interrelation among the gap components. This criterion is described further in the following.

After all of the gaps' components have been collected or listed, the next step is to see the interrelation among these gaps' components from the perspective of the architecture domains. This component interrelation assessment makes use of the already available viewpoints in order to see the relationship among components across multiple architecture domains. This step starts from the highest level of the enterprise architecture. If there are no gaps' components in the business architecture, then the step is performed in the next lower level architecture: application architecture, and so on.

```
For each gaps' components in the business architecture,  
  Trace and list all of its supporting application components  
  For each of the supporting application components,  
    Match if the application component is the gap of application architecture  
    List matched gaps' components in application architecture  
  
  For each matched gaps' components in application architecture,  
    Trace and list all of its supporting nodes  
    For each of the supporting nodes,  
      Match if the node is the gap of technology architecture  
      List matched gaps' components in technology architecture
```

Considering the nature of enterprise architecture where architectural components could be interrelated across multiple layers and could support more than one other architectural elements, many-to-many relationship is possible. Likewise, a gap component, for example an application, might support more than one gap components, for example two business services. Therefore, a situation where one gap component belongs to more than one group interrelated gaps' components is possible.

One group interrelated gaps' components could be defined as one single line of related gaps' components across multiple layers (business architecture, application architecture and infrastructure architecture) that serves a common business (change) goal.

Step 3: Filter out the model

The third step is to filter out the initial model of the architecture to only include the related gap components. The motivation of this step is to have a clearer and better picture of the fragmented of the enterprise architecture that is being analyzed. The filtered model would only consist of the affected architectural components (gap components). As the initial enterprise architecture of an organization is highly complex with many components and relationships among them, not all of the components and relationship are related to the gap components analysis. Therefore, having the filtered model of the architecture would simplify the analysis and overview.

As previously performed in step 2, change goals of the organization have been identified and linked to the gaps components. Therefore, this step also filters out the model with regards to the goals and requirements. Goals and requirements mapping could be performed by using the goals and requirements realization viewpoint.

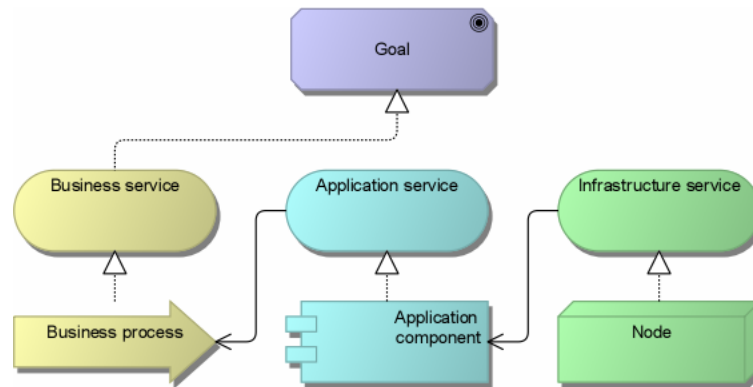


Figure 61: Filtered model for Gaps Components Group

Following the work of Buschle & Quartel (2011), the normalized model would be the simplification of all three architectural layers with the goals of the organization (motivation layer). The above figure shows the normalized model that would be referred to be used in performing this step.

The output of this step is the filtered-out enterprise architecture which covers only the related gaps components, including mapping between the gaps' components group and the goals of the organization. Therefore, each of components in gaps group could be traced to the goals of the organization. A gap component could be directly linked to the requirement that realizes the goal. Thus, this gap component can also be seen as having direct relation to goal. When a gap component does not have direct relation to requirement and goal, then the process of tracing up to its higher level of architecture is needed. The goal and requirement mapping is then performed. The mapping process goes on into the higher level of architecture until a requirement and/or goal is found.

Step 4: Calculate the importance of the components' group to the organization

The fourth step is to calculate the importance level of the gap components to the business, which is represented by the organization's goals. The step would be performed within the group that the gap components belong to. The motivation of this step is to analyze whether the gap components is highly critical, and thus highly important, to the organization. When a gap component is more important to the organization as compared to another gap component, then that particular gap component would be given more attention (higher rank of priority).

This step estimates the level of importance of each of the gaps components to the organization, or goals in general. After each of the components' level of importance has

been identified, then the level of importance of the gaps components as a group will be calculated by taking the average value of the individual components. This step makes use of the proposed method of Bedell (1985) where the importance variables have to be determined based on the perceived importance in obtaining the strategic goals of the organization or business process. The score of importance level of gaps components (e.g. business process, application, and infrastructure) ranges from 0-10.

The calculation step of the importance level of the gap components to the organization is highly subjective. It is due to the fact that the values of the importance level are derived according to subjective evaluation of related stakeholders or owner of the components. The figure below shows the guidance in scoring the importance of business process to the organization. In order to determine the importance level score for application and infrastructure, comparable decision diagram could be made, according to the work of Bedell (1985).

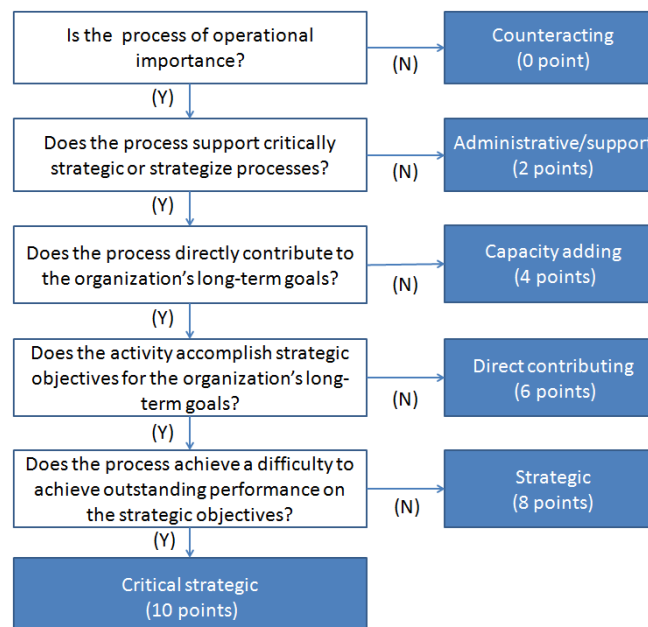


Figure 62: Determining strategic importance scores (based on Bedell, 1985)

Step 5: Calculate the effectiveness of the components' group to the organization

The fifth step of the gaps portfolio valuation is to calculate the effectiveness level of the gap components to the organization. The motivation of this step is to assess whether the gap components are performing effectively or not in accomplishing their tasks. A gap component with a lower level of effectiveness, when compared to another gap component, is perceived to be given more priority or attention when it comes to closing the gaps.

By referring to Bedell's original work, the effectiveness here refers to the effectiveness of a single information system to the activities (ESA). The effectiveness of the gaps' components to the organization is also scaled as absent (0) to high (0-10) done by IS

management in cooperation with the business organization based on their perception of the cost-effective, technical quality, and functional appropriateness. Key Performance Indicators (KPIs), measurable performance indicators, can be defined for each business goal to measure the effectiveness of IT in supporting a business goal. Therefore, the effectiveness level of the gap components could be derived by comparing the actual performance of the components to the targeted performance, which is normally stated as the service level agreement of a particular component.

$$\text{Effectiveness level} = \frac{\text{Actual effectiveness of performance}}{\text{Targeted effectiveness of performance}}$$

The work of Iacob & Jonkers (2009) on quantitative analysis of enterprise architecture could be referred when determining the performance of architecture. Architectures can be described from different viewpoints resulting in different views on architectural models. As for the performance aspects of a system, a number of viewpoints can be discerned which result in different yet related performance measures, as shown in table below.

Table 7: Viewpoints on architecture performance (Iacob & Jonkers, 2009)

View	Stakeholders	Performance measure	Description
User/customer view	Customer, application user	Response time	Time between issuing a request and receiving the results. Processing time plus waiting time
Process view	Process owner, operational manager	Completion time	Time required to complete one instance of a process which involves multiple customers, orders, products, etc.
Product view	Product manager, operational manager	Processing time	Amount of time that actual work is performed on the realization of a certain product or result
System view	System owner, system manager	Throughput	Number of transactions or request completed per time unit
Resource view	Resource manager, capacity planner	Utilization	Percentage of the operational time that a response is busy. Highly utilized components could lead to potential bottleneck

Similar to step 4, step 5 also identifies the level of effectiveness for each of the gaps' components. Then, the value for gaps groups' effectiveness level is measured by calculating the average value of the individual components within that group.

Step 6: Calculate the interdependency level of the components' group

The sixth step is to analyze the interdependency level of the gap components to the other architectural components. The interdependency level of the gaps' components is important to assess the impact or risk of changing the components. For example, when a gap component has many relationships with other components, then changing, or even removing, this component would result in other components being affected by the action.

The higher the interdependency level of the gap components, the more complex and dependent the components are.

This is done by assessing the number of relations coming in and going out from the corresponding component. The relationships that are considered in this step are for example used-by, realization, flow, trigger etc. This is done due to the nature of these relationships which are used to model temporal dependencies between behavioral concepts. The coming-in relationship to a component shows that the particular component is being dependent on the other components. The going-out relationship from a component shows that the particular component is having influence on the other components. The higher number of the coming-in and going-out relationships of a component means a higher level of inter-dependency and thus should be given more attention.

This step borrows the concepts from object-oriented systems where fan-in (coming-in relationship) and fan-out (going-out relationship) relate to interactions between objects. The fan-out of a module is the number of its immediate subordinate modules. High fan-out indicates a high degree of interdependency, and the higher the fan-out of an object, the poorer is the overall system design. The fan-in of a module is the number of its immediate super ordinate modules. High fan-in contributes to a better design of overall system since it shows that an object is being used extensively by other objects. The figure below illustrates the example of the interdependency level of gap components with regards to the other components in the enterprise architecture.

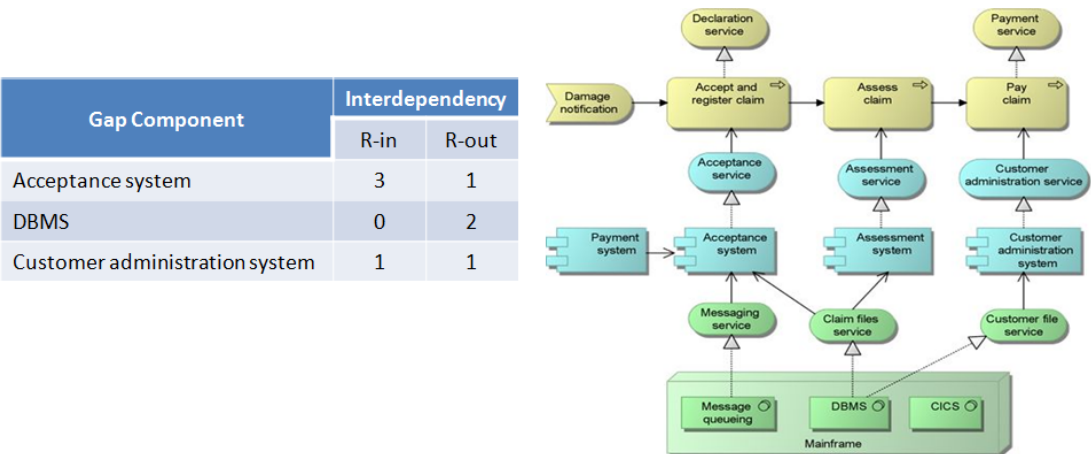


Figure 63: Example of interdependency level of gap components

Step 7: Prioritize gaps' components groups

The last step of the gaps portfolio valuation is to prioritize gaps components groups. The objective of the approach is to come up with list of prioritized gap components that are grouped together with certain business goals. The step is performed by taking into account all of the three indicators which have been previously assessed: importance level, effectiveness level and interdependency level. In order to come up with the overall score

for each gap group, the following formula is used.

$$\text{Overall Score} = (\text{Average Importance} - \text{Average Effectiveness}) + \text{Average Interdependency}$$

The average value is used in calculating the overall score because a gap group might consist of several gap components. Therefore, it is reasonable to consider the average value of each indicator to represent the gap group. The average value is also used because it could be seen as a normal distribution of the indicators' value among the gap components within a single group.

In calculating the overall score, as shown in the formula above, the approach subtracts the average importance level of the gap group with the average effectiveness level of the gap group. The approach takes the initial idea of the work of Bedell (1985) in assessing the information system to the organization. An information system, or in this case a gap component or gap group, should have a balanced value on both importance and effectiveness. The difference between the two is to represent the status of the gap group. For example, a component which is very important to the organization (has a very high score for importance level) is performing not so effectively (has a low score for effectiveness level), would have a big difference between the two values. When this happens to a gap component, then the gap component would receive higher rank for attention. Therefore, the bigger the difference between the importance and the effectiveness level of the gap component, the higher the priority is.

The absolute value of the difference between the importance and effectiveness level is not considered in this approach because the approach is only interested in assessing the importance of the component and compares it to the effectiveness level and not the other way around. For example, a gap component with low score of importance has a very high score of effectiveness. Thus, this also makes the difference between the two become high. In this case, priority would be given because the component is already performing really well and moreover, the component plays little contribution to the goals realization (small value of importance).

After performing the steps as briefly explained previously, a table showing the assessment for each of the gaps' components could be constructed. Each row represents a gaps' component consisting all of the information related to the corresponding component. Domain refers to the architecture domain the component belongs.

Table 8: Gaps components assessment

Gap Component	Domain	Goal	Importance (IBO, IAB)	Effectiveness (EAB)	Interdependency	
					R-in	R-out

For each of the groups of gaps' components, a separate table could be developed where a total score is derived from all of the components that belong to the gaps' components group. After each group has been assessed for their total score, a prioritization could then be performed.

Table 9: Group gaps components assessment

Gap Component	Importance (IBO, IAB)	Effectiveness (EAB)	Interdependency	
			R-in	R-out
Average Score				
Overall Score				

A matrix could be developed to map and prioritize these gaps components according to these dimensions.

4.4 Summary

This chapter has discussed the selected problems and the proposed solutions to address them. Two problems/limitations have been selected: aggregating a relation problem and consolidated gaps, solution and dependencies matrix. The problems selection process was mainly based on the researches that are currently performed and the feasibility of this study with regards to time allocation and knowledge acquired and required.

Aggregating a relation problem results from a condition that the relation between architectural elements, for example application components, is not captured by the current definition or specification of ArchiMate while describing the plateau concept. The cross-aspect dependency of ArchiMate metamodel limits the aggregation relationship only to core elements of architecture and leaves the relation out of the plateau definition. Therefore, the relation between the core elements cannot be drawn by referring to the metamodel because aggregating a relation to a plateau is not supported.

The Consolidated Gaps, Solutions, and Dependencies matrix is used as a planning tool when creating the work package. The purpose is to review, consolidate and integrate the gap analysis results from the Business, Information Systems, and Technology Architecture, which are performed in phase B, C and D respectively. Assessing the implications with respect to potential solutions and inter-dependencies is also the intention of the step.

Although the TOGAF framework provides illustration of how the end result of the matrix should look like, it lacks of detail guidance of how to create the matrix to come up with work packages population. Valuating the gaps is one of the objectives while developing the matrix, which later could be used as the foundation to make gaps portfolio in order to prioritize which gaps are needed to be closed first in order to realize the target architecture.

With regards to the first selected problem, two types of solutions are proposed: conceptual and practical solutions. The conceptual solution to the plateau concept is to extend the metamodel

definition of plateau that it could also aggregate the relation between the core elements. It is also recommended that the functionality of plateau duplication should be in place. The plateau definition view should be in place to define and display the architecture view which consists of core architectural elements along with the relevant relationship among them. The last solution, which is less related to aggregating a relation problem, is to introduce gap concept view to show more relationship information between the removed and added architectural elements from two plateaus.

With regards to consolidated gaps, solutions and dependencies matrix, an approach or process to gaps portfolio valuation is proposed. The process consists of seven steps, which are collecting the gaps' components, grouping interrelated gaps' components, normalizing the model by associating the business goals and requirements, calculating the importance of the gaps' components group to the organization, calculating the effectiveness of the gaps' components group to the organization, calculating the interdependency level of the gaps' components and prioritizing the gaps' components based on the three dimensions.

5 Demonstration

This chapter discusses the next step of the DSRM of Peffers et al. (2007): to demonstrate the use of the artifacts. The artifacts in this research are the solutions provided in the previous chapter. Demonstration is seen as a way to prove that the solution works by solving one or more instances of the problem. The implementation of how to use the artifact to solve the problem is then shown. The process will be applied by means of a case study in measuring the efficacy of the artifact in solving the problem. The chapter is organized as follows. Section 5.1 discusses briefly the case study method of the chapter. Section 5.2 presents a case study to demonstrate the proposed solutions. Finally, section 5.3 summarizes the chapter.

5.1 Case Study Method

The case study demonstration involves one case study: ArchiSurance. ArchiSurance case study (Jonkers et al., 2012a) is widely used in the subject of EA and it is a fictitious case developed by BiZZdesign BV and Novay, and its predecessors Telematica Institut and Telematica Research Center, in order to represent an enterprise comprehensively so that the concepts used in enterprise architecture could be applied. ArchiSurance case study is a required material to be used as an example throughout accredited ArchiMate training courses. Brief description of ArchiSurance case study has been presented in section 3.2 to show the possible and identified limitations of enterprise architecture framework, language and tool in terms of enterprise architecture transformation.

The case study is well documented by The Open Group which could be accessed publicly. Thus, document analysis of the case study documentation is then performed as the base approach in understanding the context the case. The information and assumptions used in the original form of the case are kept unchanged. Apart from document analysis, another method used in doing the case study is having internal consultants' sessions. The sessions served multiple purposes. Besides gathering insights on the problems or limitations faced by the users in dealing with enterprise architecture transformation process, consultants could be referred to have better understating on the case study description. This is possible because the consultants contributed as the writers of the case studies being used for the demonstration.

In order to operationalize the proposed solution, some inputs are needed for quantification purpose. Some of the required inputs might not be available on the original cases' set of information. Therefore, necessary additional data, information or assumptions are added to have a clearer demonstration of the proposed solutions. This method is important in order to see the process of how the proposed solution is executed.

5.2 ArchiSurance Case Study

ArchiSurance is a fictitious company that is the result of a merger of three previously independent insurance companies, which are the original ArchiSurance (the largest of the three, offered homeowner's and travel insurance), PRO-FIT (a company specializing in car insurance),

and LegallyYours (a small company specializing in legal and insurance). This section describes the background information of the ArchiSurance, its transformation overview and the demonstration of how the solution could be applied in ArchiSurance situation.

5.2.1 Case Description

The merged ArchiSurance is formed to maximize the advantage of numerous synergies between the three organizations, because they are all having similar business models. They also sold the insurance products to their customers directly through the web, email, telephone and postal mail channels. The merger discussion began when the lead investors of the three companies realized the threat of lower-cost competitors' entrance to the market. In addition, there were new opportunities in high-growth regions and each company needed significant new IT investments to remain competitive. The merger of the three companies was believed to simultaneously control the costs, maintain customer satisfactions, and take advantage of emerging markets with high growth potential.

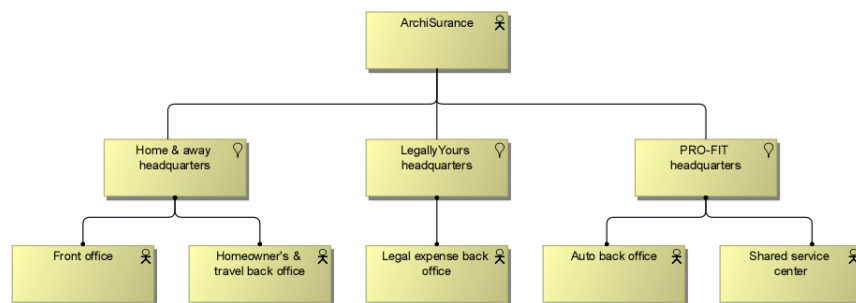


Figure 64: ArchiSurance Organization Structure (Jonkers et al., 2012a)

A result of the merger, a number of integration and alignment challenges for the new company's business process and information systems is faced. The applications landscape within ArchiSurance has become scattered and increasingly complex. This situation has led to information silos which could become more and more problematic, since they refer to the same information being stored and processed at different locations. Data redundancy is one of the problems identified besides functional overlap and non-standard communication between application instances. The above challenges result in internal instabilities, increased application maintenance costs, and obstacles to sharing information across the company and with partners. Consequently, a sizeable backlog of work requests is inevitable to be faced by IT department. This backlog had become the main concern of the top management; especially an unmet need to share information automatically with high-volume contracted sales partners and influential insurance consultants.

ArchiSurance initiated a project which aimed to clean up the application landscape. The project to rationalize the application landscape is going to be performed by conducting:

- Migrating to the integrated back office suite. The suite supports the functions of policy administration and financial transactions. The suite will consist of:
 - AUTO-U, an automated underwriting system that generates proposals and

- o policies.
 - o P-ADMIN, a packaged policy administration system that integrates with the automated underwriting system to issue, modify and renew policies. The system will also handle customer accounting and billing.
 - o VERSA-CLAIM, a packaged claim system with screens and workflow that can be configured to support ArchiSurance's three lines of business.
 - o P-CONFIG, a product configurator management used to define all insurance products, and expose these definitions to AUTO-P, P-ADMIN and VERSA-CLAIM through web services.
 - o BRIMS, a business rule management system (BRMS) consisting of a rules repository, a processing engine, a rule development environment, and an authoring tool for rule management user interfaces. The business rule engine exposes rule execution capabilities to AUTO-U, P-ADMIN, VERSA-CLAIM, and P-CONFIG through web services.
- Migrating to the strategic CRM system

It should be noted that the above initiative should be performed on the condition that all changes are invisible to ArchiSurance customers and partners. The insurer's products and services must not be affected and all customers and partners interactions must proceed uninterrupted and unchanged. The overview of the transformation process with regards to the different architectural domains is described in the next section.

5.2.2 ArchiSurance Transformation Overview

This sub-section shows the overview of the baseline architecture and target architecture of ArchiSurance company. In order to have a general overview of the as-is ArchiSurance enterprise architecture, Figure 65 below could be referred. The figure incorporates the breakdown of the business goals as well as the requirements and principles needed to support them. These upper layer concepts are originated from motivation extension.

The Figure 66 depicts how the enterprise architecture of the ArchiSurance company should look like in the future after the transformation is reached. The goals and requirements of the company are also shown and linked to the respective architectural components; could be business process or application service and other components.

In order to get focus on the gaps components for the transformation, the Figure 67 below could be referred. Only the architectural components that would be modified in the transformation process (removed, added or modified) from baseline to the targeted situation are shown.

The general overview of the gaps between the target architecture and the baseline architecture could be seen in the Figure 67 below where all of the architectural components are placed into a single canvas model. Differences between plateaus could be mapped out and displayed. The orange colored concept blocks in the Baseline architecture reflect that these components are going to be removed from the Target architecture. It means that these components are no

longer valid when the target architecture is in place (valid from the perspective of time). The green colored concept blocks in the Target architecture reflect that these components are going to be added in the architecture. It means that these components are new and valid when the target architecture is in place (valid from the perspective of time).

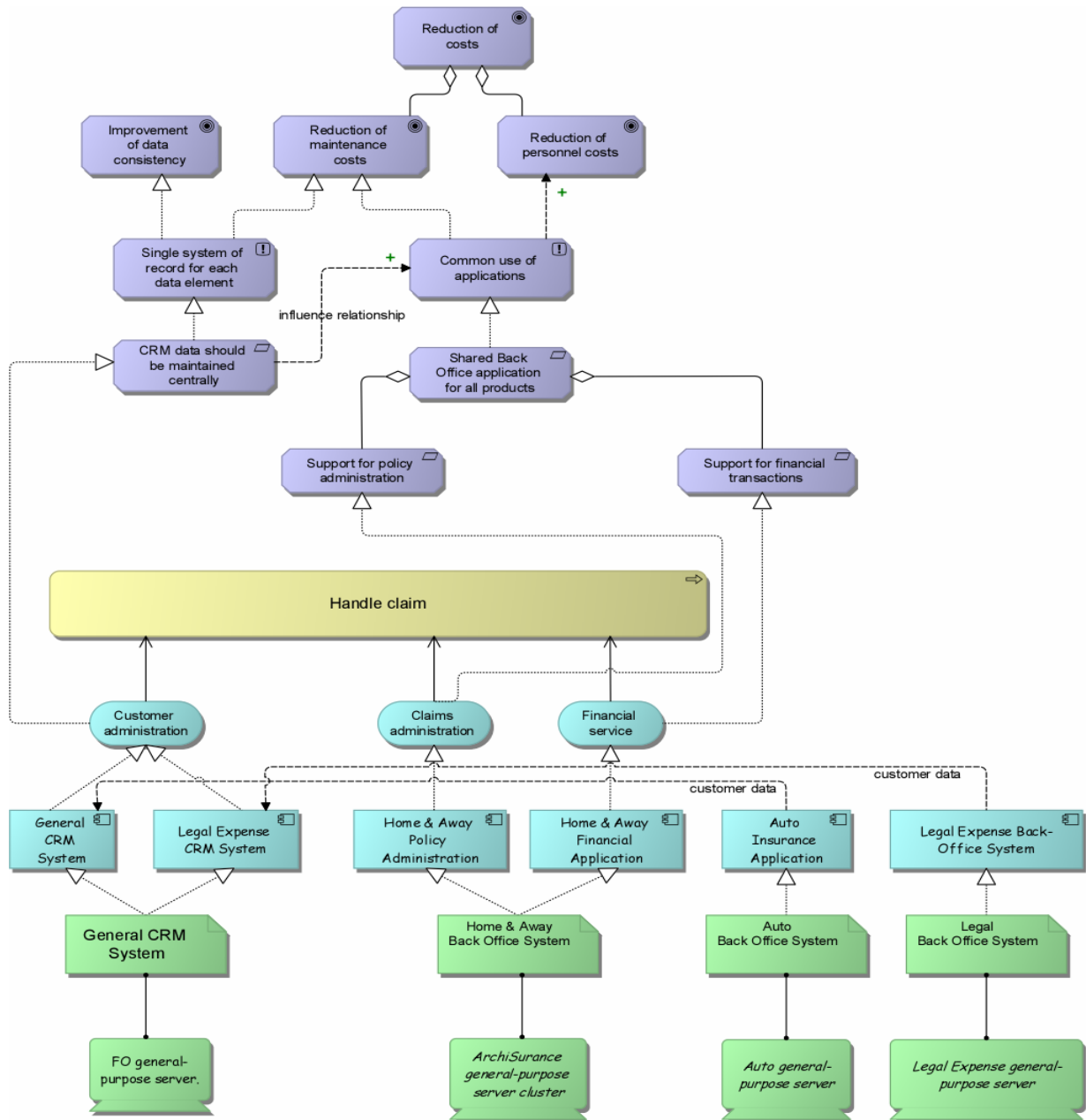


Figure 65: Overview of ArchiSurance's as-is architecture

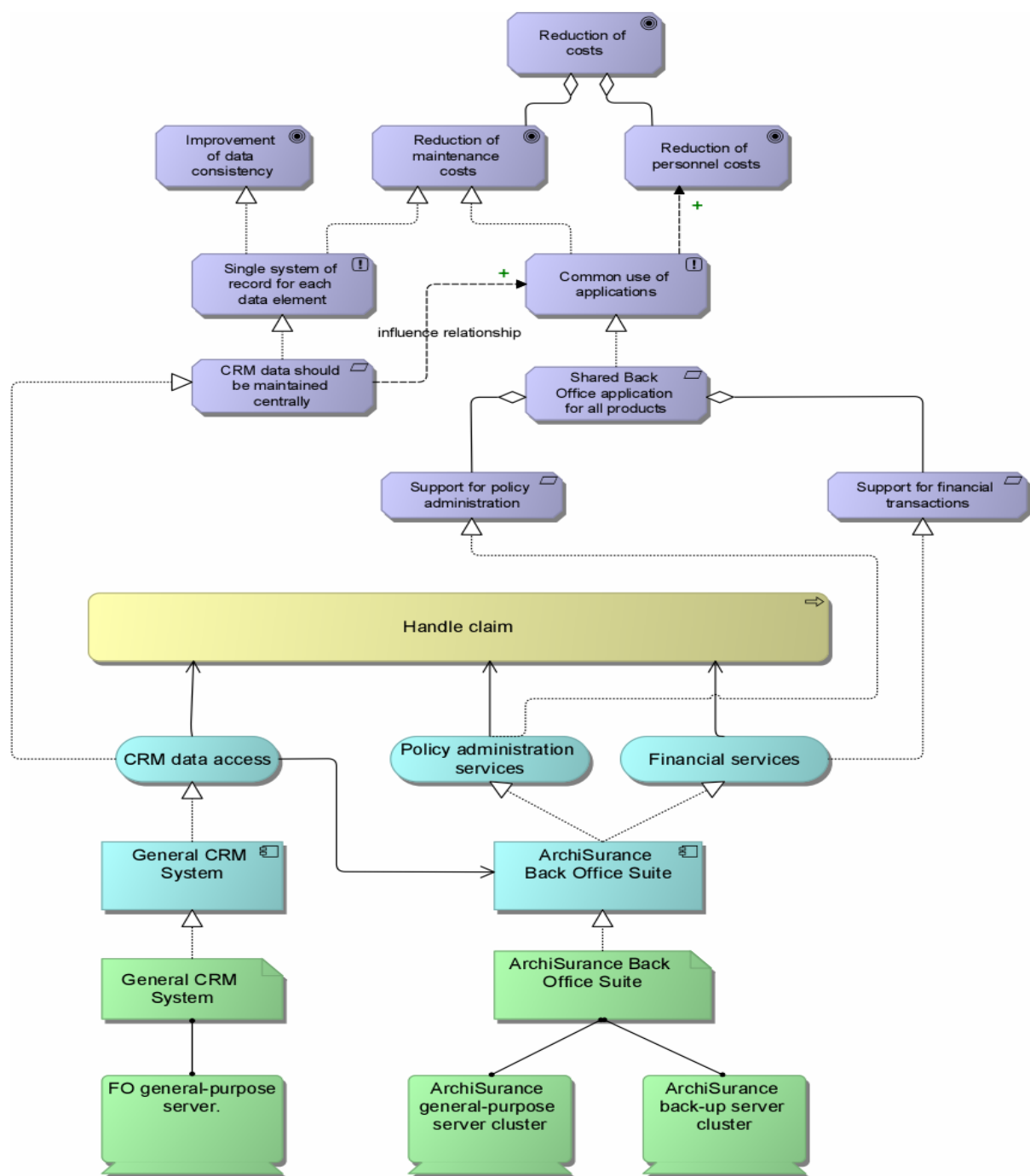


Figure 66: Overview of ArchiSurance's future architecture

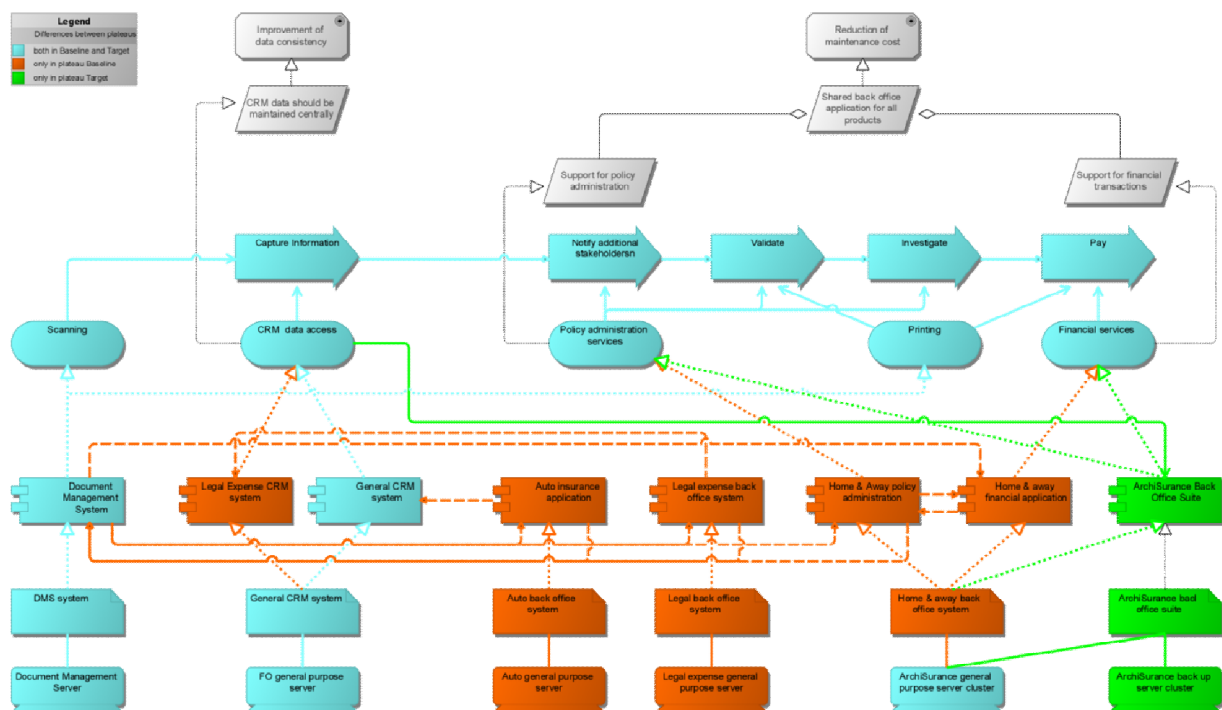


Figure 67: General overview of gaps

As for each of the architectural layers, the Figure 68 could provide general overview of what components belong to baseline architecture, target architecture or both. It is worth to note that the ArchiSurance case does not change or modify its business layer because the company wants to keep the transformation invisible to the clients or customers. Thus, business architecture would not be shown.

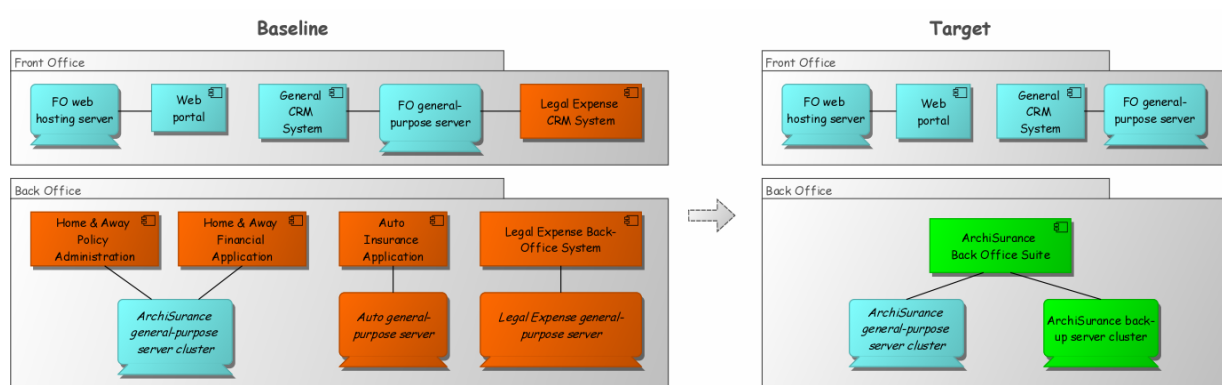


Figure 68: Affected architectural components in the transformation process

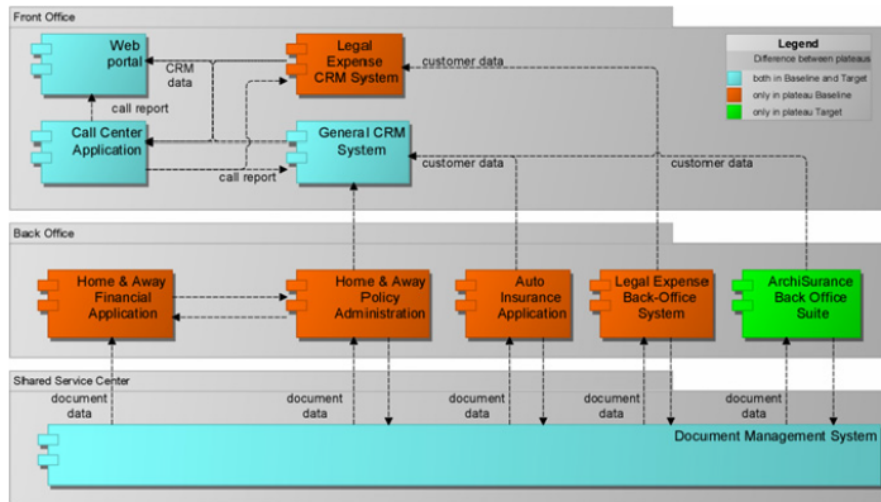


Figure 69: Overview of application architecture differences of ArchiSurance

The figure above shows the changes or the differences between the baseline architecture and target architecture of application layer. In the future, Legal expense CRM system will be removed and its functionality will be later handled by General CRM system. As for the back office applications, integration will be performed in order to merge all of the previous applications. ArchiSurance Back Office suite will be introduced as a replacement for the removed applications. Home & Away Financial application, Home & Away policy administration, Auto insurance application, and Legal expense back office system are removed.

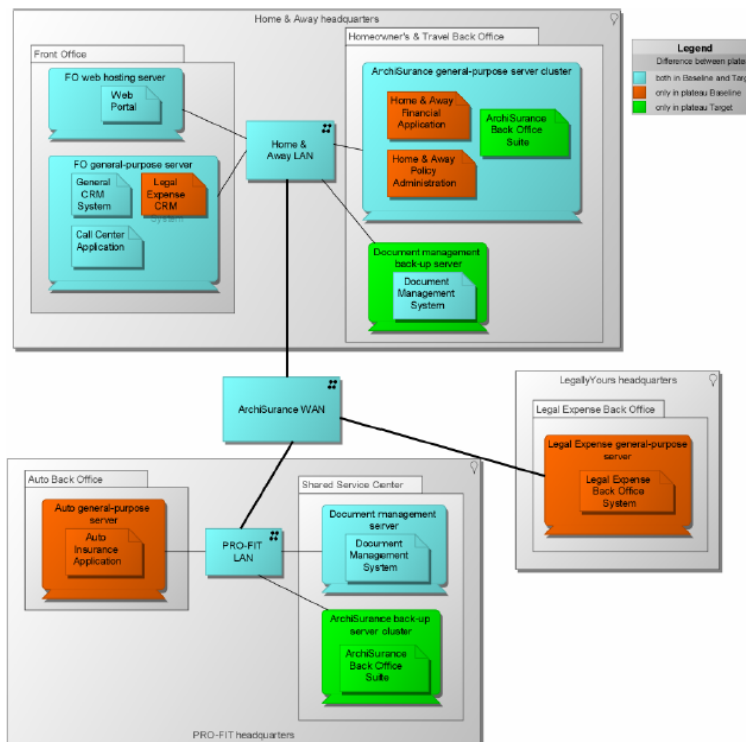


Figure 70: Overview of technology architecture differences of ArchiSurance

The transformation of ArchiSurance would also affect the technology layer. Since some of the application will be removed, the dedicated and respective servers or nodes would also be affected. Auto general purpose server and Legal expense general purpose server will be removed and replaced by ArchiSurance general purpose server cluster. In addition, ArchiSurance back-up server cluster will be added to back up the performance of the main server.

5.2.3 Solution Implementation

This section aims to demonstrate the proposed approach of Gaps Portfolio Valuation method in order to come up with the prioritized gaps groups. The figures that display the gaps between the baseline and target architecture of the ArchiSurance company could be used as main reference. The figures include the general overview of the gaps as well as the individual architectural layers. Please note that Business domain (layer) is not represented here because in the ArchiSurance case study, no changes are experienced during the transformation process.

Step 1: Collect gaps' components

The function to identify, and thus collect, the gaps' components between two plateaus has been made available in the current BiZZdesign Architect tool. The function is in the form of script, which can be found inside the "Scripts" folder on the tool installation drive. The script is part of the roadmap-menu functionalities. Below lines of script is the snapshot of the script functionalities (coding) in order to identify and collect the components that are different between the two plateaus.

```
. . . . .

forall o in modelobjects if (o.hasAttr("belongsTo")) {
    if (o.attrValue("belongsTo").contains(pA))
        SperAProf.add(o);
    if (o.attrValue("belongsTo").contains(pB))
        SperBProf.add(o); }

doubles = List();
SperProf = SperAProf + SperBProf;
SperAggreg = SperAAggreg + SperBAggreg;

forall o in SperProf if (SperAggreg.contains(o))
    doubles.add(o);

. . . . .
```

Only the components that are identified as modified and found differently in the two plateaus would be taken into account. The components could be modified, removed or added within the transformation process. The transformation process could be perceived as the difference between two plateaus, in this case is between Baseline architecture and Target architecture.

The result of the first step could be seen in below table.

Table 10: Collected gaps components

Gap Component	Domain	Remarks
Legal Expense CRM System	Application	Removed
General CRM System	Application	<i>Modified</i>
Home & Away Policy Administration	Application	Removed
Home & Away Financial Application	Application	Removed
Auto Insurance Application	Application	Removed
Legal Expense Back Office System	Application	Removed
ArchiSurance Back Office Suite	Application	New
Auto General Purpose Server	Technology	<i>Modified</i>
Legal Expense General Purpose Server	Technology	Removed
ArchiSurance General Purpose Server Cluster	Technology	<i>Modified</i>
ArchiSurance Back Up Server Cluster	Technology	New

Step 2: Group interrelated gaps' components

By going through the logic of the pseudo-code given in section 4.3.1 previously, there are nine (9) gaps groups identified. Since there is no gaps component in the business domain, then the searching starts at the application layer. The nine gaps groups are:

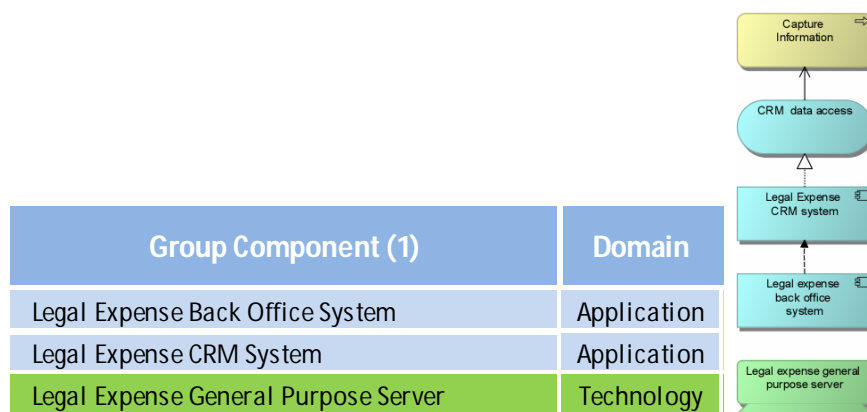


Figure 71: Group 1 – gap components

Group Component (2)	Domain
Auto Insurance Application	Application
General CRM System	Application
Auto General Purpose Server	Technology

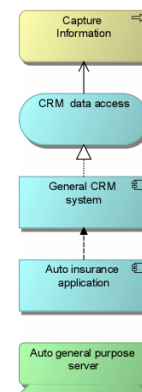


Figure 72: Group 2 – gap components

Group Component (3)	Domain
Home & Away Policy Administration	Application
ArchiSurance General Purpose Server Cluster	Technology

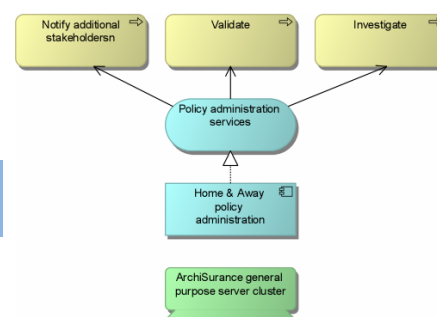


Figure 73: Group 3 – gap components

Group Component (4)	Domain
Home & Away Financial Application	Application
ArchiSurance General Purpose Server Cluster	Technology



Figure 74: Group 4 – gap components

Group Component (5)	Domain
General CRM System	10

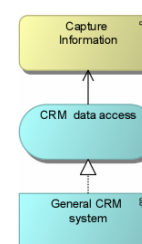


Figure 75: Group 5 – gap components

Group Component (6)	Domain
ArchiSurance Back Office Suite	Application
ArchiSurance General Purpose Server Cluster	Technology

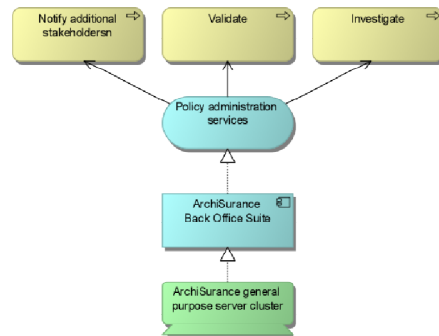


Figure 76: Group 6 – gap components

Group Component (7)	Domain
ArchiSurance Back Office Suite	Application
ArchiSurance Back Up Server Cluster	Technology

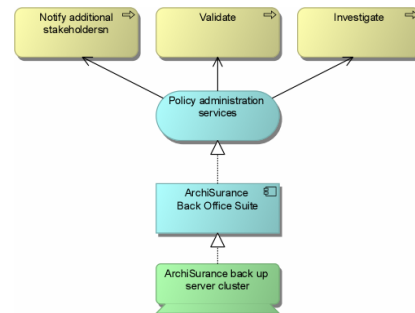


Figure 77: Group 7 – gap components

Group Component (8)	Domain
ArchiSurance Back Office Suite	Application
ArchiSurance General Purpose Server Cluster	Technology

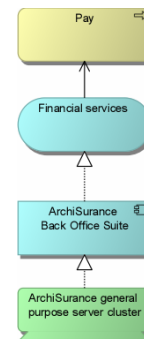


Figure 78: Group 8 – gap components

Group Component (9)	Domain
ArchiSurance Back Office Suite	Application
ArchiSurance Back Up Server Cluster	Technology

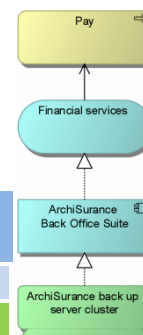


Figure 79: Group 9 – gap components

Step 3: Filter out the model

Referring back to the main figure of the general overview of the gaps between the baseline and the target architecture, it could be seen that there are two main change goals of the organization. These two goals drive the organization in transforming its architecture. The two goals are: improvement of data consistency and reduction of maintenance cost.

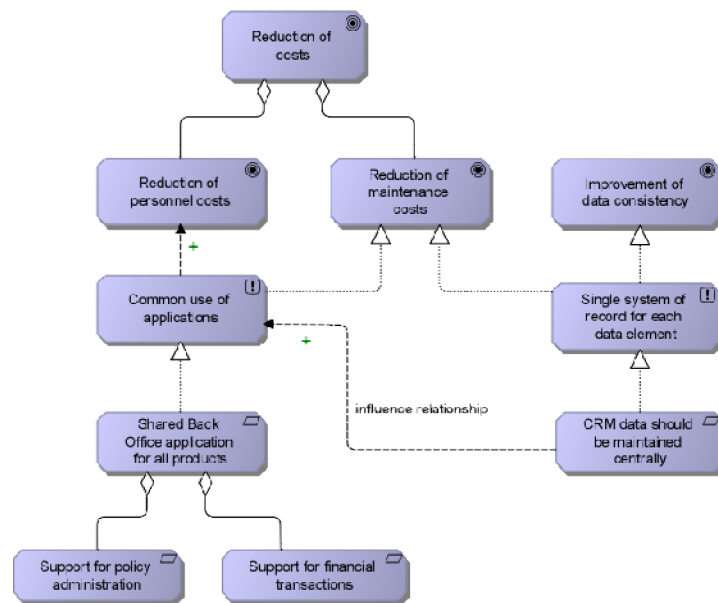


Figure 80: Fragment of goal, principles and requirements of organization

The goals are further broken down into sub-goals. The principles of the organization, from motivation extension perspective, are also taken into account, which realize the organization's goals and/or sub-goals. Some requirements are also identified in order to reach the goals and at the same time, being complied and aligned with principles held by the organization. The figure above shows the fragment of the figure 61 and figure 62 that only deals with the goals, sub-goals, principles and requirements.

Table 11: Gaps components – Goals Mapping

Gap Component	Requirement / Goal
Legal Expense CRM System	Data consistency
General CRM System	Data consistency
Home & Away Policy Administration	Cost reduction
Home & Away Financial Application	Cost reduction
Auto Insurance Application	Data consistency
Legal Expense Back Office System	Data consistency
ArchiSurance Back Office Suite	Cost reduction
Auto General Purpose Server	Data consistency
Legal Expense General Purpose Server	Data consistency
ArchiSurance General Purpose Server Cluster	Cost reduction
ArchiSurance Back Up Server Cluster	Cost reduction

In order to filter out the model, the approach only considers the upmost goals of the organization that are connected or linked to or realized by the gaps components. By referring back to the normalized model template, which is introduced and used in Buschle & Quartel (2011), the following filtered model is resulted.

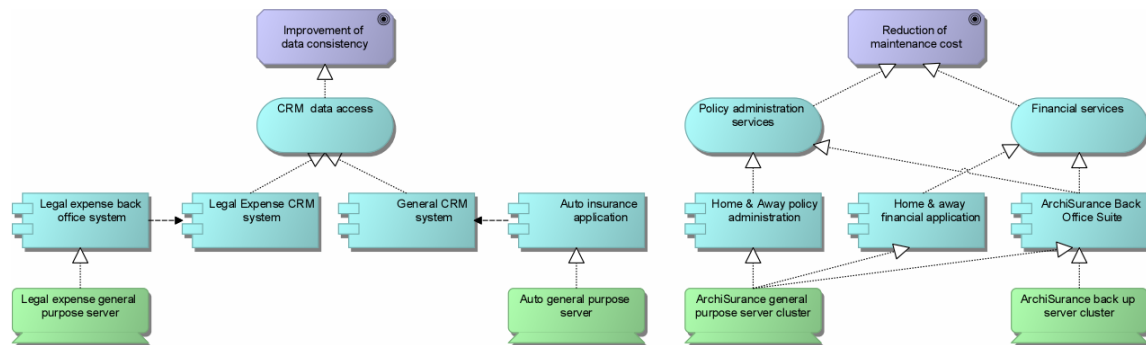


Figure 81: Filtered-out model of gaps components & goals

Step 4: Calculate the importance of the components to the organization

As previously mentioned in section 4.3.1, the importance variables have to be determined based on the perceived importance in obtaining the strategic goals of the organization or business process. It means that a system (or an architectural element) is considered strategically important when the activities supported are crucial to the organization or business process in obtaining its strategic objectives.

The score of importance level of gaps components (e.g. business process, application, and infrastructure) ranges from 0-10 with 0 means not important and 10 means very (strategic) important. For this case study demonstration, it is assumed that the values of the gaps components' importance level are already available. The values are assumed to have been discussed and decided through dedicated and series of meetings with the relevant stakeholders, such as top management level and owners of particular architectural components.

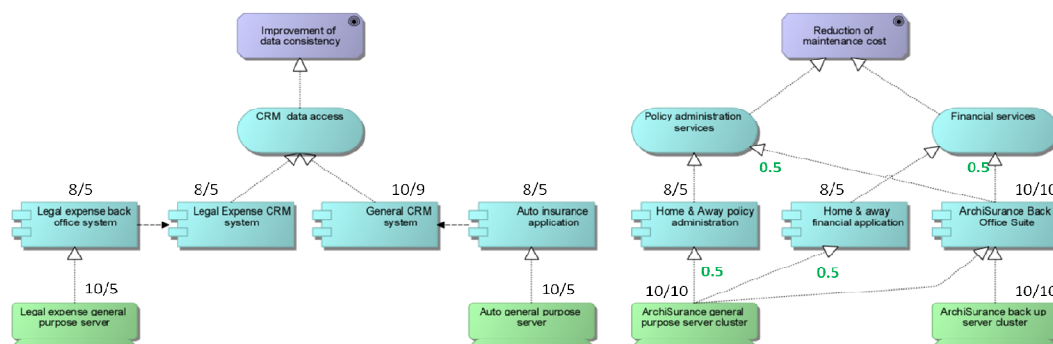


Figure 82: Allocated level of importance and effectiveness of gaps components

Table 12: Gaps components level of importance

Gap Component	Importance (IBO, IAB)
Legal Expense CRM System	8
General CRM System	10
Home & Away Policy Administration	8
Home & Away Financial Application	8
Auto Insurance Application	8
Legal Expense Back Office System	8
ArchiSurance Back Office Suite	10
Auto General Purpose Server	10
Legal Expense General Purpose Server	10
ArchiSurance General Purpose Server Cluster	10
ArchiSurance Back Up Server Cluster	10

Step 5: Calculate the effectiveness of the components to the organization

Similar to importance level of components, the level of effectiveness of components needs to be calculated by the subject matter experts as well. The owner of components is the one responsible for maintaining and monitoring the performance of the components under his/her area. In section 4.3.1, several measurements could be used to assess the effectiveness level of certain components. Each components, then, has its own way of measuring the effectiveness or performance depending on the nature of the component and also on the architectural domain.

For example, utilization could be used to measure the performance of components in the technology domain, such as server. Processing time, on the other hand, could be used to measure the performance of the application component at application domain to measure the amount of time that actual work is performed on the realization of a certain product.

The level of effectiveness of component to the organization is measured by comparing the target performance to the actual performance of the component. Therefore, this could also be inferred as by how much percentage the KPI of certain component has been reached. The score of effectiveness level of gaps components (e.g. business process, application, and infrastructure) ranges from 0-10 with 0 means not effective and 10 means very effective, or in other words, it meets the performance goal.

For this case study demonstration, it is assumed that the values of the gaps components' effectiveness level are already available. The values are assumed to have been calculated by the relevant stakeholders, such as owners of particular architectural components, through certain measurement approach. The table below shows the values of effectiveness level of the gaps components in supporting the organization's performance.

Table 13: Gaps components level of effectiveness

Gap Component	Effectiveness (EAB)
Legal Expense CRM System	5
General CRM System	9
Home & Away Policy Administration	5
Home & Away Financial Application	5
Auto Insurance Application	5
Legal Expense Back Office System	5
ArchiSurance Back Office Suite	10
Auto General Purpose Server	5
Legal Expense General Purpose Server	5
ArchiSurance General Purpose Server Cluster	10
ArchiSurance Back Up Server Cluster	10

Step 6: Calculate the interdependency level of the components

In order to calculate the number or the level of interdependency of the gaps components, the approach refers to the overall architecture that includes all of the necessary relationship among the components. Therefore, the approach does not consider only the relationships among the gaps components but also the relationships of the gaps components to other architectural components.

To do this, the figure below will be referred. The figure is the fragment of Figure 63 as displayed previously by omitting the motivation extension concepts that include goals, sub-goals, principles and requirements. It is due to the fact that calculating interdependency level of the gaps components does not require motivation extension concepts. Only the relationships between the gaps components and the other architectural components are considered.

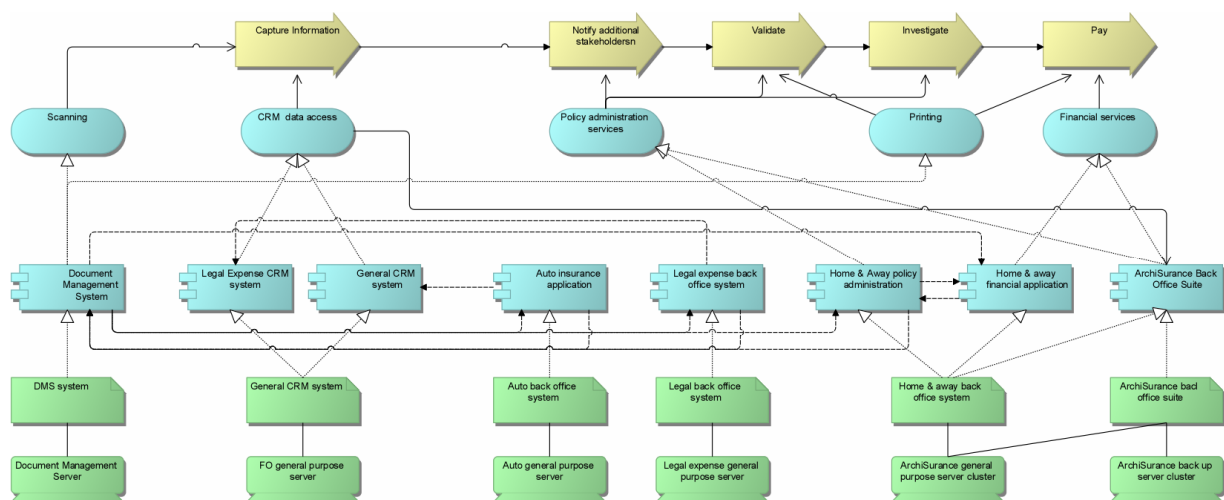


Figure 83: Interrelations of the architectural components

The table below shows the list of the interdependency level of each gaps components to other components in the architecture. For example, Legal Expense CRM system has two incoming (R-in) relationships: (1) is the realization relationship from the technology service concept, called General CRM system and (2) is the data flow relationship from Legal Expense Back Office system application. This means that the Legal Expense CRM System application is being dependent on two architectural components. The Legal Expense CRM system has one outgoing (R-out) relationship which is the realization relationship to the CRM data access application service. This means that the existence of Legal Expense CRM system affects one architectural component.

Table 14: Gaps components' level of interdependency

Gap Component	Interdependency	
	R-in	R-out
Legal Expense CRM System	2	1
General CRM System	2	1
Home & Away Policy Administration	3	3
Home & Away Financial Application	3	2
Auto Insurance Application	2	2
Legal Expense Back Office System	2	1
ArchiSurance Back Office Suite	2	2
Auto General Purpose Server	0	1
Legal Expense General Purpose Server	0	1
ArchiSurance General Purpose Server Cluster	0	3
ArchiSurance Back Up Server Cluster	0	1

Step 7: Prioritize gaps' components

The last step is to calculate the overall score for each of the gaps groups. The following formula is used in order to calculate the overall score:

$$\text{Overall Score} = (\text{Average Importance} - \text{Average Effectiveness}) + \text{Average Interdependency}$$

The overall score is assessing the average score for each of the indicators' values: importance, effectiveness and interdependency. The subtract (difference) between the average importance and average effectiveness of the component is considered crucial and relevant. This is due to the fact that the component's performance must be balanced with its importance.

A high value of difference between importance and effectiveness of a component means that there is a big gap in the performance level where the component is considered important to the organization. For example, a component with importance level 8 and effectiveness level of 3 (difference of 5) will be perceived as a more important gap that needs to be closed when compared to another component with importance level of 8 and effectiveness level of 7 (difference of 1). The latter component could be concluded as having a balanced importance

and effectiveness level. When both components need to be transformed, then the earlier component will have more priority to be transformed. The following table shows the overall score for all of the gaps groups that have been identified and calculated.

Table 15: Overall scores of gaps groups

Group Component (1)	Importance	Effectiveness	Interdependency	
			R-in	R-out
Legal Expense Back Office System	8	5	2	2
Legal Expense CRM System	8	5	2	1
Legal Expense General Purpose Server	10	5	0	1
Average Score	8,666666667	5	2,67	
Overall Score			6,33	

Group Component (2)	Importance	Effectiveness	Interdependency	
			R-in	R-out
Auto Insurance Application	8	5	2	2
General CRM System	10	9	2	1
Auto General Purpose Server	10	5	0	1
Average Score	9,33	6,33	2,67	
Overall Score			5,67	

Group Component (3)	Importance	Effectiveness	Interdependency	
			R-in	R-out
Home & Away Policy Application	8	5	3	3
ArchiSurance General Purpose Server Cluster	10	5	0	2
Average Score	9	5	4	
Overall Score			8	

Group Component (4)	Importance	Effectiveness	Interdependency	
			R-in	R-out
Home & Away Financial Administration	8	5	3	2
ArchiSurance General Purpose Server Cluster	10	5	0	2
Average Score	9	5	3,5	
Overall Score			7,5	

Group Component (5)	Importance	Effectiveness	Interdependency	
			R-in	R-out
General CRM System	10	9	1	1
Average Score	10	9	1	
Overall Score			2	

Group Component (6)	Importance	Effectiveness	Interdependency	
			R-in	R-out
ArchiSurance Back Office Suite	10	5	1	2
ArchiSurance General Purpose Server Cluster	10	10	0	1
Average Score	10	7,5	2	
Overall Score			4,5	

Group Component (7)	Importance	Effectiveness	Interdependency	
			R-in	R-out
ArchiSurance Back Office Suite	10	5	1	2
ArchiSurance Back Up Server Cluster	10	10	0	1
Average Score	10	7,5	2	
Overall Score			4,5	

Group Component (8)	Importance	Effectiveness	Interdependency	
			R-in	R-out
ArchiSurance Back Office Suite	10	2,5	1	2
ArchiSurance General Purpose Server Cluster	10	10	0	1
Average Score	10	6,25	2	
Overall Score			5,75	

Group Component (9)	Importance	Effectiveness	Interdependency	
			R-in	R-out
ArchiSurance Back Office Suite	10	2,5	1	2
ArchiSurance Back Up Server Cluster	10	10	0	1
Average Score	10	6,25	2	
Overall Score			5,75	

Gaps Group	(Importance - Effectiveness)	Interdependency	Overall Score
3	4,00	4,00	8.00
4	4,00	3.5	7.50
1	3.67	2.67	6.33
8	3.75	2,00	5.75
9	3.75	2,00	5.75
2	3,00	2.67	5.67
6	2.5	2,00	4.50
7	2.5	2,00	4.50
5	1,00	1,00	2.00

CRM Integration → (Data Consistency)

Group 1, Group 2, Group 5, Group 6, Group 7
 $6.33 + 5.67 + 2.00 + 4.50 + 4.50$

23.00

Back Office Integration → (Maintenance Cost)

Group 3, Group 4, Group 8, Group 9
 $8.00 + 7.50 + 5.75 + 5.75$

27.00

Referring back to the goals of the transformation of ArchiSurance, which are to reduce the cost and to improve the data consistency, the gaps groups could be differentiated. The gaps group 1, 2, 5, 6 and 7 are related to the realization of the data consistency goal, which should be realized by integrating the CRM system. On the other hand, the gaps groups 3, 4, 8 and 9 are related to the realization of maintenance cost reduction, which should be realized by integrating the back – office applications. The figure below depicts the distribution of the gaps groups on the 2-axis layout: (1) importance & effectiveness and (2) interdependency.

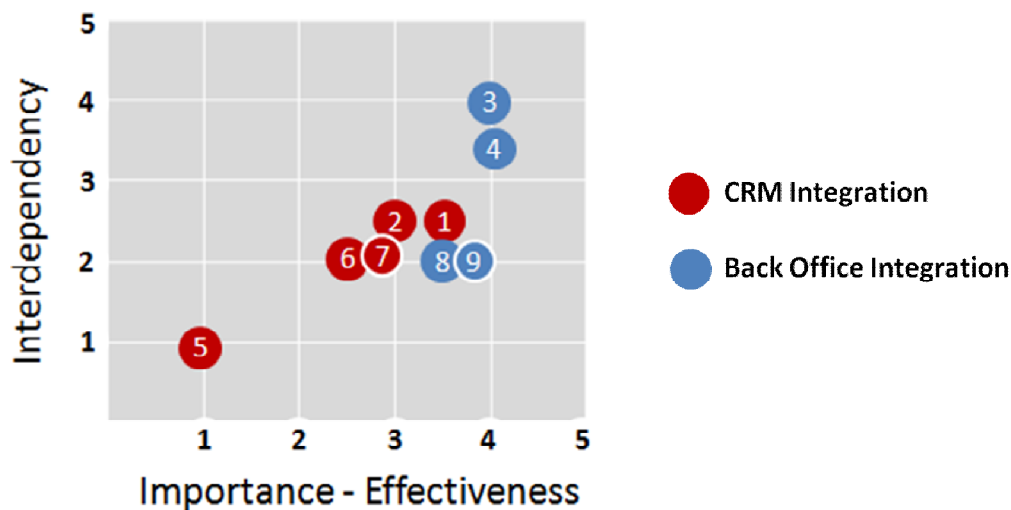


Figure 84: Gaps' groups mapping

The end result of the gaps portfolio valuation suggests that gaps groups that contribute to the Back-Office integration realization have more overall score than the gaps groups that contribute to the back-office integration realization. Therefore, for the ArchiSurance to transform and reach the target architecture, the transformation path that integrates the Back-Office system first (the blue dot path) is recommended.

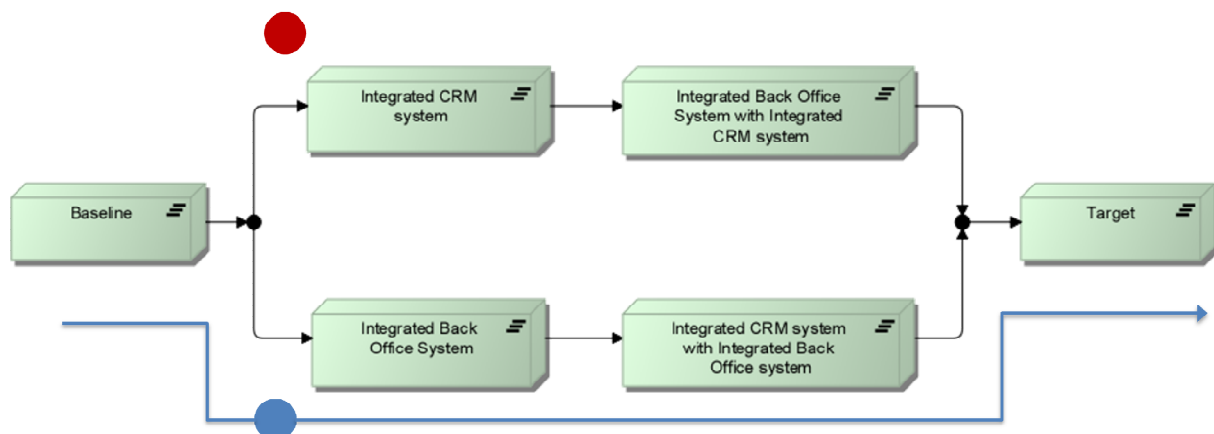


Figure 85: Alternative transformation paths

The rationale behind the gap groups' prioritization is that the organization would have more information on deciding which transformation path to take. Closing the most prioritized gap groups, in accordance to the organization's goals, is crucial especially in a situation when an organization, later in the future, runs out of resources to continue the transformation path. By selecting the high important and urgent projects (closing the gaps group with highest priority), then the initial investment by the organization would (hopefully) deliver most significant impact to the organization.

5.3 Summary

This chapter has demonstrated the implementation of the gaps portfolio valuation approach in a means of a case study. ArchiSurance case study is chosen as the case study example. Due to the merger of three companies, the organization encounters performance issues and cost concerns. The case has two transformation objectives or goals, which are to reduce the maintenance cost and to improve the data consistency. The maintenance cost reduction could be reached by integrating the back office system. The data consistency improvement could be reached by integrating the CRM system.

The demonstration of the approach involves all of the seven steps within the approach. However, it should be noticed that several steps, especially step 3 and 5, are performed in a simplistic way by taking assumptions that several information are readily available. Through the execution of all of the steps, the gaps portfolio valuation approach recommends ArchiSurance to integrate the CRM system first, in order to reach the target architecture.

By applying the method, it is hoped that the organization would gain more insightful information to decide and ensure that the most important (urgent) gaps are closed first. This would result significantly different in a situation where the organization lacks of the resources in continuing the transformation process and thus, not closing other gaps.

6 Evaluation

Evaluation phase is the fifth part in the design science research methodology. It involves observing how well the artifacts support a solution to a problem. According to DSRM of Peffers et al. (2007), the evaluation could be in the form of comparison of the artifact's functionality with the solution objectives, the results of satisfaction surveys, clients' feedback or simulations. This chapter presents a qualitative analysis in the form of interview in order to evaluate the artifacts of the thesis. The chapter is organized as follows. Section 6.1 discusses the dimensions of evaluation that would be used to assess the artifacts. Section 6.2 describes the interview approach used in the evaluation phase, including the setting of the interview, the targeted respondents and the interview questions scripts. Section 6.3 provides the analysis and the results of the evaluation interview collected. Finally, section 6.4 summarizes the chapter.

6.1 Evaluation Dimensions

There are two main artifacts that are being evaluated in this research: the solution to aggregating a relation problem and the gaps portfolio valuation approach. Interview will be used as the means of evaluation according to several dimensions. The interview description would be elaborated in the later section. DeLone & McLean (2003) provides six dimensions and measures when evaluating the Information Systems (IS) success, as depicted in the figure below.

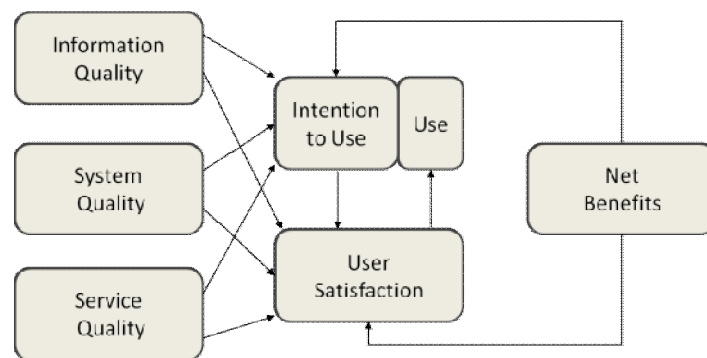


Figure 86: IS Success Model (DeLone & McLean, 2003)

1. System Quality – measures of the information processing system itself. It may include several indicators, such as adaptability, availability, reliability, response time and usability.
2. Information Quality – measures of information system output. This dimension could cover indicators such as completeness, ease of understanding, personalization, relevance and security.
3. Service Quality – measures of the overall success of the information system department. Indicators within this dimension include assurance, empathy and responsiveness.
4. Use and intention to use – measures recipient consumption of the output on an information system. This includes nature of use, navigation patterns, number of visits and number of transaction executed.

5. User satisfaction – measures recipient response to the use of the output of an information system. The indicators within this dimension could be repeat purchases, repeat visits and user surveys.
6. Net Benefits – measures the impact to the use of the information system. The indicators that could represent the net benefits of an artifact could be cost savings, expanded markets, incremental additional sales, reduced search cost and time savings.

According to DeLone & McLean (2003), information quality and system quality are the most important quality components for evaluation dimensions to measure the success of a single system. Service quality, on the other hand, measures the overall success of the information system department. Thus, for this reason, service quality would not be considered to evaluate the artifacts.

Due to the fact that both of the proposed artifacts have not been used nor tested independently by the real users or the potential stakeholders, use, intention to use and user satisfaction dimensions would not also be measured and evaluated. The demonstration of the artifact could not be considered as the usage of the artifact in real situation with the real users.

Net benefits dimension is one of the dimensions being evaluated qualitatively through the interview. It is to see the use impact, benefits or advantages of the artifact in practice. Therefore, three dimensions of DeLone & McLean (2003) IS success model would be used in the evaluation session: information quality, system quality and net benefits. Several indicators criteria as well as questions as the realization of the dimensions would be provided in the interview question script section.

The evaluation of the first artifact, the solution to aggregation to a relation problem, would be performed by assessing whether the idea of adding relations aggregation to plateau concept is correct and complete. As mentioned in this research, a relationship is normally done between a concept (from/to) and another concept. The proposed solution is to extend the ability of ArchiMate modeling language to be able to draw a relationship from a concept to a relationship between concepts. Looking at this initiative from the perspective of the expert in modeling language is needed.

The second artifact, the gaps portfolio valuation, would be evaluated by looking at the general and overall steps in the proposed approach or method. The evaluation session is intended to see the completeness of the proposed approach from the perspective of the expert as well as the consultant that deals with the issue and with real context in practice.

6.2 Interview

Evaluation phase of DSRM of Peffers et al. (2007) intends to evaluate or assess the methods or artifacts which could be performed by collecting feedback from potential end-users via survey and from the client. Interview could also be one of the means to collect feedback with regards to artifacts' evaluation. Interview is an example of qualitative analysis for qualitative data with the emphasis on making sense or understanding a phenomenon.

Gall, Gall & Borg (2003), as cited in Turner (2010), classifies three formats of interview: informal conversational interview, general interview guide approach, and standardized open ended interview. The informal conversational interview is performed in a spontaneous and impromptu approach by questioning in a natural interaction setting with the participants. The general interview guide approach provides a more structured interview approach by having the interview questions beforehand. However, the execution of the interview is fully dependent on the interviewers and therefore, is semi-structured and lacks of consistency. The standardized open-ended interview is the most structured among the three. All of the wordings of the interview questions are well prepared and the interview sessions are conducted in identical way. This means all of the questions being asked during the interview are the same in all of the interview sessions. Also, the way the interview is performed, such as the flow or the interview structure, is also held identical.

In this research, the evaluation session uses the semi-structured interview format. The format has been selected due to the practicality and flow of the evaluation session. As later depicted in interview setting, the flow of the session is really flexible where respondents could jump into questions while research introduction is being performed. The conversation might already cover the essence and answers to some of the interview questions even before the question and answer session has been started. However, all of the main points (dimensions/indicators) are ensured discussed.

6.2.1 Interview Setting

Below table shows the structure or schedule of the interview session. Each interview session is conducted in maximum 90 minutes. There are three interview sessions in order to evaluate the artifacts.

Table 16: Interview setting

Duration	Activities
45 minutes	<ul style="list-style-type: none"> • Research introduction. • Solution and method (artifacts) presentation. • Method demonstration.
30 minutes	Interview question script. Question and Answer session based on the scripted interview questions.
15 minutes (optional)	Open discussion. Intended to gather additional and necessary comments, feedback and suggestion for improvement.

The participants or respondents to the interview sessions in this research are selected based on their expertise and role in the area of the research. Three representatives are selected, as follow:

1. Ontological/Semantic expert; also referred to as interviewee 1. The ontological expert holds an important role in providing feedback from the expert's perspective. The unique

and extensive knowledge on ontological area that is possessed by this respondent would allow the evaluation session to assess the validity of the proposed artifact, especially the conceptual solution to the aggregation to a relation problem.

2. Enterprise architect expert; also referred to as interviewee 2. Similar to ontological/semantic expert, this representative holds an important role in providing feedback from the expert's perspective. The extensive knowledge about enterprise architect, including ArchiMate modeling language, which resides within this respondent, would assess the correctness and completeness concerns of the proposed artifacts.
3. Internal consultant; also referred to as interviewee 3. This representative deals with clients directly in daily practice. The internal consultant representative aims to provide perspective on the relevance and the usability of the proposed artifacts, based on their extensive experience with clients. This type of respondent would be able to provide feedback on the artifact's pragmatic ease of use and understanding.

6.2.2 Interview Question Script

Several questions have been constructed to address the dimensions selected for the qualitative interview sessions to evaluate the artifacts. Follow up questions are also constructed for several questions in order to extract more information based on the answer to the main questions. The table below lists all of the constructed questions to evaluate the two artifacts. The questions are derived from the selected dimensions of the evaluation, which are further classified into several criteria or indicators.

Table 17: Interview questions – Evaluation sessions

#	Dimensions	Criteria	Question
Artifact 1: Conceptual solution to aggregation to a relation problem			
1	System quality	Correctness	From the theoretical (formal) perspective, does the solution provide sufficient level of correctness? What needs to be improved?
2	System quality	Feasibility	Do you think it is feasible for the approach to be included in the ArchiMate specification?
3	Net benefits	Usefulness	Do you think this solution is useful in practice? Why?
Artifact 2: Gaps portfolio valuation			
4	Information quality	Completeness	Based on your experience, does the method include all the required activities in practice?
5	Information quality	Ease of understanding	Is the method clear? Which steps need further elaboration?
6	System quality	Feasibility	Do you think it is feasible for the approach to be implemented?

7	Net benefits	Usefulness	Do you think this method is useful in practice and give an insight to the user in managing their architectural gaps?? Why?
---	--------------	------------	--

6.3 Analysis and Result

The result of the interview sessions are analyzed and discussed in this section. The complete transcript of the answers to the interview questions, as well as the additional comments and feedback could be found in appendix D. The following table visualizes the summary of the evaluation session.

Table 18 : Summary of evaluation session

#	Evaluation Criteria	Interviewee 1	Interviewee 2	Interviewee 3
Artifact 1: Conceptual solution to “aggregation to a relation” problem				
1	Correctness	<i>Somewhat OK.</i> The current solution might lead to ternary relationship confusion. Treating relationship as a class concept could be an idea to approach this better.	<i>OK.</i>	<i>OK.</i>
2	Feasibility	<i>Somewhat OK.</i> It is not a minor change, a bit intrusive.	<i>OK</i>	<i>Somewhat OK.</i> The change is fundamental change, but needed. Discussion on The Open Group forum would give clearer picture.
3	Usefulness	<i>OK</i>	<i>OK</i>	<i>OK</i>
Artifact 2: Gaps portfolio valuation				
4	Completeness	<i>Somewhat OK.</i> Quite a challenge to validate the numbers.	<i>Somewhat OK.</i> Calculation is hard to judge.	<i>OK.</i> It should be a combination. What works in theory does not always work in practice. Other external factors should be considered, such as laws enforcement, management insights, etc. Coming up with prioritized lists is one thing, communicating it and convincing others are other things.

5	Ease of understanding	OK. Some steps could be explained more, for example: grouping criteria, differentiating change goals.	OK.	<i>Somewhat OK.</i> Several calculations seem technical. Intensive discussion is needed with the SMEs to allocate the numbers. Weight of relationship in ArchiMate could be considered.
6	Feasibility	OK	OK. Especially when the enterprise architecture is well documented in the organization	<i>Somewhat OK.</i> The method provides transparent approach. Some management levels might not like the idea.
7	Usefulness	OK	OK.	OK. In practice, other external factors would be used in taking decisions, such as benefits, risks, costs, laws (obligation), and even insights.

In the following, for every sub-section the first part would discuss the objectives of the solutions, in other words re-describing the problems that have been identified and selected. Then, some implications would also be elaborated when the solutions are about to be implemented. The implementation of the solution could have some impacts, either to the ArchiMate modeling language or the BiZZdesign Architect tool. Finally, general description to summarize the evaluation criteria for each artifact is given.

6.3.1 Adding Relation Aggregation to Plateau Concept

The first artifact that is being evaluated is the initiative to “adding relation aggregation to plateau concept”. The conceptual problem behind the aggregating a relation problem is the limited definition of plateau concept which explicitly define that only architectural component could be aggregated to a certain plateau. This limits the fact that relations between components could not be aggregated and thus could not be captured as part of a certain plateau.

By adding relation aggregation to plateau concept, the artifact extends the definition of a plateau concept. An aggregation relationship could be drawn between the plateau concept and the relationship line between two components that belong to that particular plateau. This additional definition has also been drawn and explicitly depicted in the proposed plateau concept definition in the ArchiMate modeling language specification.

Several concerns, including implications that are resulted from the solution have been discussed previously. The elaboration about the implications concludes that all of the relationship types between two components could also be aggregated to the plateau concept,

given the condition that both of the interacting components are also part of the plateau. The source and target components of a relationship have been defined clearly in the solution description. Both components must be valid for, or belong to, a certain plateau in order to aggregate the relationship between them to the plateau.

Having said so, the first artifact solves the core problem comprehensively because it allows the aggregation relationship to be directly link the relationship between components to the plateau. This additional aggregation relationship fulfills the basic need of the problem. The artifact also does not require many modifications to the existing ArchiMate metamodel with regards to the implementation and migration extension. The only adjustment needed is the additional part of having an aggregation relationship from the relationship between components to the plateau concept. There is no new type of relationship being introduced. However, the concept aggregating a relationship to an object is never introduced or implemented before. Aggregation relationship has always been used to express that some objects might gather themselves to form a bigger unification object. Thus, aggregation relationship is always dealing with objects with other objects. Moreover, the artifact is very feasible to be implemented in the existing enterprise architecture tool. In fact, the existing version of BiZZdesign Architect is able to perform this functionality.

The followings are the summary of the interview respondents' answers with regards to the first artifact. The explanation follows the categorization of the dimensions being evaluated.

System quality

- *Correctness* - Different perceptions are given by the respondents. According to Enterprise Architecture expert and Internal Consultant, the proposed artifact offers sufficient level of correctness. It is easy to understand and is aligned with the current specification. The proposed idea is already correct and it is needed in practice.

On the other hand, according to Ontology/Semantic expert the proposed artifact could be further improved. The main objectives of the solution are well understood. However, the proposed model could be seen as ternary relationship, which does not support the goal. In semantic modeling, ternary relationship is something that needs to be avoided due to its complexity in terms of cardinality problems. In order to handle this perception, having relationship as a class is recommended. The proposed improvement would be further elaborated in the recommendation section of Chapter 7. Moreover, the Ontology/Semantic expert questions the use of aggregation relationship to represent that certain components belong to certain plateau. According to his expertise, aggregation relationship between plateau and component and aggregation relationship, for example, between component and other components are semantically different. Architectural components do not belong to architecture state, they exist in the state. In conclusion, while introducing new construct for plateau is possible, introducing new relationship type to represent plateau should also be possible.

- *Feasibility* - The respondents agree that the initiative is feasible to be implemented in the

ArchiMate specification. Understanding the need and motivation of the solution, even though the initiative might be considered new and does not exist before, it is something to apply for. The current specification of ArchiMate does not allow aggregation between relationships. While, in practice it is need. The fact that BiZZdesign Architect tool has already implemented the initiative shows that the solution is feasible and the tool is developed and built by a better modeling principle. It is however good to take note that the solution proposes some fundamental changes to the language. Having a relation between relations is something new to the language concept and thus, has quite an impact. Further discussion is recommended to bring this matter to the Open Group forum.

Net benefits

- *Usefulness* – The respondents agree that the proposed artifact is useful and could be used in different scenarios. For example, to detail out the flow relationship between applications in order to put more information on what kind or what type of information are being exchanged. Another example of the application of the solution is about capability extension where some components, arranged together in particular way, realize a capability. By applying the proposed artifact, the respondents agree that it would help the users in making the architecture information clearer and more explicit. The users might be able to select which relationship(s) between two components that belong to a plateau is (are) also part of the plateau.

6.3.2 Gaps Portfolio Valuation

The second artifact that is being evaluated is the “gaps portfolio valuation” approach. The artifact is presented in the form of a method on how to prioritize the gaps components with regards to business goals. The problem that is addressed by the artifact is the lack of concreteness or detailed guidance in dealing with the consolidated gaps, solutions and dependencies matrix. The matrix is introduced and processed in phase E of ADM: Opportunities and Solutions, in the section 2.3.1. The purpose of the step 3 of phase E is to review, consolidate and integrate the gap analysis results from the Business, Information Systems, and Technology Architecture, which are performed in phase B, C and D respectively. Assessing the implications with respect to potential solutions and inter-dependencies is also the intention of the step. Therefore, the main objective of the artifact is to provide additional insight in realizing the target architecture, or in other words, to prioritize the work packages that need to be implemented. The output of the artifact is the list of prioritized gaps groups to offer more perspectives in determining which gaps needed to be closed first in order to realize the target architecture.

The gaps portfolio valuation approach consists of seven steps and considers several inputs to be readily available which are resulted from the previous phases of TOGAF ADM. The available inputs are gaps analysis results of business architecture, gaps analysis results of information system (application) architecture, gaps analysis results of technology architecture. During the

demonstration phase by implementing the artifact to the ArchiSurance case study, several points could be derived as discussions to evaluate the completeness of the artifact. In the demonstration, it is assumed that the organization has well-documented and structured enterprise architecture. This enables the complete overview of the architecture and this is likely to happen only in ideal case or situation. Many organizations, in real life, still struggle depicting and documenting their enterprise architectures. Thus, the assumption should be taken further into account when implementing the artifact in real life situation.

Calculating the importance level of components is done in simplistic way. Again, the assumption is taken where organization has already assessed the level of importance of its architectural components to the business or organization's goals. The values of importance are normally gathered through interview session with the owner of the components or subject matter experts. Furthermore, when calculating the effectiveness of the components to the organization, the artifact still applies simplistic approach.

The effectiveness level is calculated by comparing the targeted performance of specific component with the actual performance of the component. Thus, the measurement for each component would be different depending on the architectural domain and the nature of the component. For example, a server component's effectiveness could be measured by calculating the throughput level of the server performance. Response time could be used to measure the effectiveness of an application component. Since both of these values, importance and effectiveness, are still assessed in simplistic ways, a more detailed approach could be further developed.

Another point of discussion is about the interdependency level of components. In this artifact, the interdependency level is a separate value or indicator for calculating the overall score of gaps groups. One may perceive the interdependency level to be also part of indicators when analyzing the importance level of components. A highly important component could be perceived to have high number of interdependency level towards other components. Since it is important, then it also influences or affects quite number of other components. The artifact considers the importance level to be associated mainly to the business goals of the organization and not to the complexity level of the component. Thus, these two values are treated separately. The followings are the summary of the interview respondents' answers with regards to the second artifact. The explanation follows the categorization of the dimensions being evaluated.

System quality

- *Feasibility* - The respondents agreed that it is feasible for the method to be applied and supported in the current Architect tool. Some of the analyses needed by the method are already supported and the functions (scripts) are available, for example identifying the gap components and identifying the incoming and outgoing relationships of a component. Moreover, some calculations, after being verified and validated, could be made automated by utilizing the script functionality within the Architect tool. This way, it would be even more convenient for the users in performing the proposed method.

Information quality

- *Completeness* – The respondents agreed that the proposed method covers the necessary steps within the specified scope, which is on gaps components. The identified steps represent the general and logical steps in performing the objective of the method. Step 5, when calculating the effectiveness of the gap components to the organization's goals, could be elaborated more. Knowing that different component types on different architectural domains require different types of measurement, the method should also provided a more detailed guideline on how to perform these measurements independently. Taking into consideration the relationships weights in ArchiMate could be useful while measuring the level of interdependencies among components. The relationship weights are defined in the ArchiMate specification for the purpose of having derived relationships between two concepts or components in the architecture. By analyzing the weights, relationships are treated according to their impacts or influences towards other components within the architecture.
- *Ease of understanding* – The respondents agreed that steps in the proposed method are understandable and easy to follow. However, Step 2 (grouping the interrelated gaps components based on goals) is somehow confusing and could be clearer. More explanation might be needed when depicting the relationship between goals and gap components. Better illustration and motivation would help the users tracing the goals-gap components relation. Explicit differentiation between as-is goals and change goals would help the users understand the relation between the (change) goals and gaps components. More practical guideline on grouping criteria would allow the users to perform the steps on their own.

Even though the steps of the proposed method are logical and easy to follow, when it comes to judging the correctness of the method, several considerations are noted. Firstly, it would be a challenge by itself to evaluate the validity of the numbers being assigned to the effectiveness and importance level of the gaps components. The current demonstration assumes that the values (or numbers) have been measured previously and thus, available. Secondly, the way the gap components are grouped needs more validation. The method should establish a clear guideline of what constitutes a group. Moreover, since the group is done based on goals, there should be clear distinction between the as-is goals and the change goals that motivate the transformation efforts.

Net benefits

- *Usefulness* – The respondents agreed that the proposed method could help the users making decisions with regards to gaps prioritization. Even though we could never be sure that by implementing the recommendation resulted from the method would deliver the best outcome, it is always nice to have more information and perspectives. The respondents agreed to the motivation of the gaps prioritization. Closing the highest gap priority would help the organization transform itself, especially in a situation where future investments (organization's resources) are uncertain.

6.4 Summary

This chapter evaluates the two proposed artifacts by having the semi-structured interview format with the related experts in enterprise architect, business practitioner and ontology/semantic area. In general, the overall evaluation shows positive feedback in most of the aspect being evaluated which also indicates that applying the two proposed artifacts would lead to a positive contribution in practice. Several considerations are also given by the experts for potential future improvement, for example introducing relationship as a class in ArchiMate, and separating as-is goals from change goals. More comprehensive demonstration and evaluation approach (with more interview sets) could improve the validity level of the benefits of the proposed artifacts.

7 Conclusion

This final chapter concludes what have been presented in the research. Section 7.1 summarizes all of the research questions that have been addressed throughout the study. Section 7.2 outlines the two types of contributions of the research: theoretical (academic) contribution and practical (industrial) contribution. The limitations of the research are discussed in section 7.3 which later motivates the recommendations for future research, as presented in section 7.4.

7.1 Reviewing the Research Questions

The main goal of the research is to further improve the development process of a roadmap plan for enterprise architecture transformation. To meet this goal, the main research question is formulated as follow:

“How can EA transformation be improved in the form of a roadmap plan?”

In order to provide structure in answering the main research question, the main research question is further divided into three research questions. The remainder of this section will summarize the answers to these questions.

RQ1: *What do the Enterprise Architecture frameworks say about the transformation process?*

- *Who is the key user of roadmap plan and what is the main function of roadmap plan according to this key user?*
- *What are the step-by-step guidelines in developing the roadmap plan?*

These questions are discussed and answered in Chapter 2. The main objective of these questions is to have basic understanding on the research topic. Identifying the key user of the roadmap plan is essential in shaping the direction of the proposed solution. This is imperative so that the outcome of the research would provide meaningful applicability in real case implementation.

The enterprise architect has been considered as the key user in terms of roadmap plan analysis and visualization. This role is responsible to ensure the architecture’s comprehensiveness since he/she integrates multiple views to each other and handles conflicting concerns. The enterprise architect must consider various stakeholders’ concerns in communicating the roadmap plan. In principal, roadmap plan is created by the enterprise architects after necessary and related inputs from different subject matter experts are collected. The subject matter experts, for example, could be business architect, application architect, technology or infrastructure architect, and program or portfolio managers.

In general, the roles of enterprise architect vary from understanding and interpreting requirements; creating useful model; validating, refining and expanding the model; and managing the architecture. To be able to perform the roles, enterprise architect must deal with

other types of architects that co-exist within the organization. Most of the time, the differentiation of architects is rather unclear and thus the possibility of mixing and overlapping the types of architects is high. Ideally, for each architecture domain, there should be a dedicated type of architect. For example, business architect will handle the business architecture domain; application architect will be dealing with the application architecture domain; and infrastructure architect will be responsible for infrastructure architecture domain.

Some competencies, which are classified into professional and personal, are required for the enterprise architect to perform the tasks effectively. Among the professional competencies are business skills and methods, enterprise architecture skills, program or project management skills, IT general knowledge skills, technical IT skills and legal environment knowledge. The professional competencies related with knowledge, attitude and skills necessary to perform successfully in a specific function or role. Personal competencies include communication skills, personality characteristics, creativity and leadership. Personal competencies are important in interacting with different kinds of stakeholders including the management or business level stakeholders and domain-specific stakeholders or technical level stakeholders.

A roadmap plan is the abstracted plan for the business or technology change, typically operating across various disciplines and over multiple years. It describes the path (or journey) of change, over a certain period of time, from the current situation (baseline architecture) to the desired situation (target architecture). This could be used as a guideline in monitoring the change process (enterprise architecture transformation) by analyzing the gap between the target and baseline architectures. Therefore, a roadmap plan is important for program management and investment decision since it holds the data about the current, under way and planned architectures which are making up the development programs of an organization.

Considering the nature of enterprise architect's work that deals with various level of stakeholders, ranging from business to technical level, it is important that the roadmap plan does not provide too many (technical) details on a single view. It is, however, required that a roadmap plan be able to further detail out the necessary (technical) information when needed. This flexibility factor is important for an effective roadmap plan communication to various stakeholders.

The second part of the first research question aims to explore the current step-by-step guideline provided by the EA frameworks in developing the roadmap plan which provides the basic foundation or information on the general guideline. This serves as the starting point or foundation to identify some problems already encountered in exploring the existing guidelines and determining whether the guideline is sufficient.

There are numerous EA frameworks in use today which might overlap in certain areas according to stakeholders' concerns. The EA framework that is extensively referred to is the TOGAF, The Open Group Architectural Framework. The framework is chosen as the main EA framework due to its wide implementation in the industry and its access openness to public. It is developed and maintained by members of The Open Group who are working within the

Architecture Forum. Thus, it may be used freely by any organization wishing to develop enterprise architecture for use within that organization.

Enterprise architecture transformation, including roadmap plan development, is described in the phase E and phase F of the ADM (Architecture Development Methods) of TOGAF. ADM is a tested and repeatable process that covers the establishment of an architecture framework, development of architecture content, transitioning and governance of architecture realization. Phase E (Opportunities and Solutions) conducts the identification of major implementation projects that are meant to realize the target architectures defined in previous phases. Phase F (Migration Planning) addresses the formulation of a set of detailed sequence of transition architectures with a supporting implementation and migration plan by analyzing the costs, benefits and risks.

Through its ADM cycle, TOGAF provides general guideline of the whole cycle of enterprise architecture development, including planning, transitioning or migrating and governing processes. TOGAF introduces necessary concepts, step-by-step processes in order to transform from baseline architecture to target architecture. Gap analysis becomes an important step to take in order to identify differences between two states of the architecture. The results of gap analysis would lead to determine what projects are needed to close the gaps and finally reach the target architecture.

To put the guideline into practice, a modeling language is needed to provide a uniform representation for diagrams that describe enterprise architectures. For this purpose, ArchiMate as the enterprise architecture modeling language is also analyzed. ArchiMate is an EA modeling language which is open and independent to enable enterprise architects to describe, analyze and visualize the relationship among business domains in an unambiguous way. ArchiMate is chosen as the modeling language being analyzed due to its evolution to be fully aligned with the TOGAF standard. It is also the product of The Open Group.

In its evolution, ArchiMate includes core concepts and two extensions to accommodate the concerns of modeling concepts for the architecture artifacts description with regards to business goals, architecture principles and requirements (motivation extension) as well as projects, programs, migration, transition architecture and gaps (implementation and migration extension). The implementation and migration extension is intentionally developed to serve the concepts and metamodels needed to describe enterprise architecture transformation.

Among the newly introduced concepts within this extension are plateau, gap, deliverable and work package. Moreover, the ArchiMate also introduces new viewpoints as a way to focus only on particular aspects of the architectures. This is to communicate only the related concerns of specific users groups, for example what to be included and what to be excluded in the view. The new viewpoints are project viewpoint, migration viewpoint and implementation and migration viewpoint. Project viewpoint is suitable to see the relation between the business goals and programs and have an overview that all business goals are sufficiently covered by the current portfolio. The migration viewpoint is used to model the transition of architecture change from

baseline to target architecture. The implementation and migration viewpoint relates programs and projects to the areas of architecture that they implement. This allows modeling the programs' scope and activities in terms of the realized plateaus or affected enterprise elements.

RQ2: *How is the Enterprise Architecture transformation currently supported by the Enterprise Architecture tools?*

- *What are the limitations of the Enterprise Architecture tools in analyzing and visualizing the Enterprise Architecture transformation?*

The second research question takes a further step in analyzing the current support provided by the enterprise architecture tools available in the industry. After finding out how the transformation process should be performed, exploration on the current support of EA tools is the next consideration. By doing so, the identification of EA tools supports' limitations or problems in terms of roadmap plan analysis and visualization is performed. These questions are discussed and answered in Chapter 3.

A selection of the existing EA tools is done by referring to the Gartner's Magic Quadrant as of October 2013. The quadrant analyzes and compares several EA tools and classifies them into four quadrants: leaders, challengers, visionaries and niche players; based on the completeness of vision and ability to execute dimensions. Three EA tools have been selected for further review: IBM (Rational System Architect), Avolution (ABACUS) and BiZZdesign (BiZZdesign Architect). These tools are selected by considering their positions at the quadrant as well as the easiness of access granted to observe them. Also, all of these EA tools are both ArchiMate 2.1 and TOGAF 9.1 certified.

Generally, the selected tools provide extensive support on enterprise architecture management. However, the support is fully concentrated in the as-is situation of the architecture, such as how my enterprise looks like at the moment. This could be the result of the fact that the maturity level of organization implementing the concept of enterprise architecture is still limited. Most organizations are still dealing with depicting their current enterprise architecture. Less support is shown in the area of enterprise architecture transformation where roadmap plan comes into analysis and visualization to move from baseline to target architecture.

IBM Rational System Architect introduces its internal concepts related and necessary to make the process of enterprise architecture transformation more pragmatic and practical, such as workspace, lifecycle states and milestones. Avolution provides the users flexibility and interactivity in managing their model of enterprise architectures. It offers component's catalogue management which could be applied in supporting work packages management. The catalogue could serve as the main source of information regarding the corresponding components to be used for further analysis. BiZZdesign Architect offers its supports which are align with both the EA framework and modeling language.

Two types of limitations or problems with regards to roadmap plan analysis and visualization are identified: practicality problems and guideline-related problems. Practicality problems are

derived from the daily usage of the EA tools which reflects the problems that are faced by the clients, or the users and in this case are the enterprise architects, especially related to the enterprise architecture transformation process and roadmap plan development. Guideline-related problems are related to the existing guidelines, provided by EA frameworks and ArchiMate. This type of problems is identified during the literature review conducted by the author of the thesis.

Among the practicality problems are aggregating a relationship problem; updating component problem (versioning); date validity checking (consistency) limitation; timeline view; and components relationship in different plateaus. Three guideline-related problems are: the implementation factor assessment and deduction matrix; the Consolidated Gaps, Solutions, and Dependencies matrix; and work package prioritization. Two specific problems have been selected out of those eight problems to be further analyzed and addressed. The two problems are aggregating a relationship problem and the consolidated gaps, solution and dependencies matrix. The considerations used while selecting the problems are the existing research currently being conducted, and the feasibility with regards to time limitation and required knowledge. The proposed solutions to the selected problems in the end should provide contributions both from academic perspective and industry perspective.

RQ3: How could the development of roadmap plan be improved in ArchiMate and Architect?

The third research question delivers the most interesting component of this thesis. This question is addressed by referring to the selected two problems, as previously identified by research question 2. The development of roadmap plan is hoped to be improved in ArchiMate by providing several solutions to the aggregating a relationship problem. The concept of plateau in ArchiMate could be improved by further expanding its definition to not only include (aggregate) architectural core elements but also the relationship among the core elements.

The current definition of plateau concept, as depicted in the implementation and migration extension metamodel, limits the users to include the relationships among architectural elements to the plateau. Although the current EA tool allows such relationship aggregation, it could be perceived as not aligned with the modeling language standard because the metamodel definition restricts that only core elements could be aggregated to the plateau. The roadmap plan, from the perspective of BiZZdesign Architect, could be further improved by taking into account the solutions offered by this thesis in addressing the first selected problem. These solutions are meant to enhance the practicality of the EA tool so that the users would take more benefits in terms of ease of use and usefulness.

Having plateau duplication functionality would help the user in defining new state (new plateaus) of the architecture because it could be understood that the new state of the architecture would share several commonalities with the previous state of the architecture with some modifications. Thus, users do not have to create and define the whole components of the architecture from the beginning. Of course, independent treatment to the newly created (duplicated) architecture must be possible.

Furthermore, having a specific a view definition for a plateau would be helpful for the users. The current EA tool does not support this functionality. The definition view enables the users to have an overview of a specific plateau. Therefore, it would be practical for the users to compare more than one plateau to see the difference between states of enterprise architectures. The plateau definition view could be an extended form of layered viewpoint which has been introduced in ArchiMate modeling language.

The current gap concept utilization could be further improved by including the relations between the affected architectural elements in two states of enterprise architecture. The components of previous state of architecture could be modified or removed. In the new state of architecture there could be new architectural elements introduced. The current EA tool does not provide the relation between the removed and the newly created components. By utilizing gap concept, users would have more information readily available concerning the affected components.

Referring back to the main research question: *“to what extent can EA transformation be formally specified in a form of a roadmap plan?”*, this thesis analysis the guideline provided by the EA framework (TOGAF) and the widely accepted standard modeling language (ArchiMate). The EA transformation specification in the form of roadmap plan, based on the analysis of the thesis, still lacks of detailed definition.

EA framework (TOGAF) has provided comprehensive guidelines through its Architecture Development Method (ADM) cycle. EA transformation is covered in phase E and phase F of ADM. In phase E, projects or work packages, which need to be implemented to close the gaps, are identified. In phase F, roadmap plan, which addresses the formulation of a set of detailed sequence of transition architecture with a supporting implementation and migration plan by analyzing the costs, benefits and risks, is developed.

From the perspective of the ArchiMate modeling language, some additional concepts are introduced to accommodate the EA transformation. The concepts are included in the implementation and migration extension of the language. Metamodel with regards to the EA transformation concepts are specified. This thesis analysis the specification of this metamodel, especially on the concept of plateau and comes up with several solutions for further improvement.

Prior to developing a roadmap plan which displays all of the work packages or implementation projects in timeline view, the gaps which would be addressed by those work packages need to be identified and sorted out. This thesis proposes gaps portfolio valuation approach. By having such prioritization, the process of sequencing work packages could take benefits from the prioritized gaps as additional consideration.

7.2 Research Contributions

The thesis makes several contributions to both theoretical and practical fields of enterprise architecture, as summarized below.

7.2.1 Theoretical Contributions

1. Assessment on guidelines for enterprise architecture transformation. The first theoretical contribution offered by this thesis is the assessment about the guidelines on enterprise architecture transformation which is provided by TOGAF. The enterprise architecture transformation part is covered by phase E and phase F of TOGAF's ADM cycle. The thesis dedicates section 2.3 for this purpose. The list of problems and limitations are summarized in section 3.2.
2. Solutions with regards to plateau concept. The second theoretical contribution is the follow up of the result of the assessment. The plateau concept is introduced in ArchiMate modeling language to accommodate the concepts or description of enterprise architecture states as defined in TOGAF. Section 4.2 of the thesis partly addresses this contribution by expressing (additional) necessary extension to a plateau concept.
3. Gaps portfolio valuation. The third theoretical contribution is also the follow up of the result of the assessment. This thesis proposes 7 steps of gaps portfolio valuation approach. The approach aims to value and prioritize the gaps which are needed to be closed by implementing projects (work packages) in order to move from Baseline Architecture to Target Architecture, or to move between two consecutive states of architectures. Section 4.3 explains each of the steps and chapter 5 demonstrates how the approach is applied in the form of case study.

7.2.2 Practical Contributions

1. Problems and/or limitations inventory. The first practical approach is the inventory of problems and/or limitations, as listed and summarized in section 3.2. This inventory could be perceived as an initial list for the internal team of BiZZdesign to further improve their tool in supporting their clients or users.
2. Solutions with regards to plateau concept. This second practical contribution is related to the selected problem: aggregating a relationship problem. This problem covers both conceptual (theoretical) and practical (EA tool) problems. Section 4.2 partly provides solutions for improvement of the BiZZdesign Architect tool. The solutions can be adapted and implemented in the current EA tool of BiZZdesign Architect. Hopefully, by implementing the solutions, BiZZdesign Architect would provide more pragmatic solutions to the users.
3. The third practical contribution could be derived from gaps portfolio valuation in helping business users have a clearer picture in analyzing and assessing the gaps. Prioritizing work packages or projects that need to be implemented to reach the target state of the architecture could be made more comprehensive by taking into consideration the prioritized gaps that would be closed by the implementation of projects.

7.3 Research Limitations

This thesis has several limitations that have been identified throughout the research process and thus, might have influenced the results of the thesis. Firstly, the scope of the thesis is limited and focused on TOGAF, ArchiMate and BiZZdesign Architect as the enterprise architecture framework, enterprise architecture modeling language and enterprise architecture tool respectively. This selection is made to accommodate the limited time assigned to accomplish the thesis. Although this set of framework, modeling language and tool is considered sufficient to represent the topic area due to their alignment and support to one another, there exist quite number of other enterprise architecture frameworks. Some frameworks are dedicated for specific area of enterprise, for example defense area or governmental area.

Secondly, out of five practical problems and three guideline-related problems, only two problems are selected to be further observed in this thesis. Again, this is mostly due to time constraint. The two selected problems are considered sufficient enough to cover both theoretical and practical perspective. Moreover, the list of the problems and limitations regarding the enterprise architecture transformation guideline and support are deducted mainly from the session conducted with the internal consultants and the observation and analysis done by the author. With the strong assumption that the users' level of awareness and maturity in this area is still limited, direct users or clients involvement is not performed. The inputs from the internal consultants are taken as the representation of the users and clients.

Thirdly, external validity (generalizability) of the proposed approach has not been tested because only through the case study the approach is demonstrated. Additionally, the case study implementation is performed more to show how the proposed approach is operationalized, and not validated. Furthermore, the case study used in the demonstration is fictitious. Therefore, the usability and usefulness of the proposed approach may differ per organization, or type of organizations (different real case studies).

Fourthly, step 4 and step 5 of the gaps portfolio valuation approach are done in simplistic way. Step 4 is dealing with calculating the importance of the gaps components to the organization and step 5 is dealing with calculating the effectiveness of the gaps components to the organization. The effectiveness of the architectural components to the organization is assumed readily available within the organization. Different measurements and/or techniques in calculating this effectiveness level for each architecture domain could be performed by utilizing and/or incorporating the work of Iacob & Jonkers (2009).

7.4 Recommendations for Future Research

Considering the above mentioned research limitations, several recommendations could be derived. These could be perceived as interesting directions for future research to be conducted either to counter-analyze or to improve and enhance the results of this research.

Exploration on other enterprise architecture frameworks could have been performed more thoroughly so that guideline comparison in the enterprise architecture transformation process could be taken in place. For example, considering the Pragmatic Enterprise Architecture Framework (PEAF) as comparison would improve the understanding of the EA frameworks' support since PEAF is made less complex than TOGAF with an appropriate level of complexity to be easily understood and used by users. TOGAF and PEAF do not compete with each other but they are rather complementary. Therefore, when having key users' perspective on how well the EA frameworks guide and support the transformation process, pragmatic framework should also be considered.

The remaining limitations and problems inventory resulted from this thesis could be further analyzed and observed. By considering several existing researches that are currently running or have been started and applying the perspective taken by this research; evaluating the analysis and visualization of roadmap plan for enterprise architecture transformation; would be interesting. This is to check and ensure that the existing researches address the limitations identified in this research.

A future research could be performed with the focus of implementing the recommendations to the plateau concept on the enterprise architecture tool, such as BiZZdesign Architect. This thesis does not fully implement all of the recommendations given. Deeper technical knowledge on the tool would be useful in exploring whether or not the recommendations are feasible and applicable to the existing tool.

As previously mentioned, the external validity of the proposed approach needs to be further evaluated. More demonstration on various case studies would improve the generalizability of the approach when different situations are in place. Furthermore, applying the proposed approach of gaps portfolio valuation in a real life case is recommended. In order to minimize the subjectivity in evaluating the result of the thesis, more internal consultants and enterprise architecture practitioners could be asked for their assessment and professional feedback.

This thesis does not consider about the maturity level of the enterprise's awareness as well as the capability on the architecture. The thesis, based on professional inputs from internal consultant, assumes that most of the enterprises still lack of awareness or maturity in the area of enterprise architecture transformation. They are more matured or well informed when it comes to depicting their as-is or existing architectures, as it is the first logical step before coming to the transformation process. Some adjustments to the steps or assumed inputs for the proposed approach of gaps portfolio valuation might be different when considering the capability and maturity level of the enterprises.

An in-depth research could be done to detail out the steps of the proposed approach of gaps portfolio valuation. Specifically, step 4 and step 5 of this thesis could be performed in a more comprehensive way. For example, calculating the effectiveness of the architectural components might make use of the work of Iacob & Jobbers (2009) which identifies different performance quantitative measurements for different architecture domains' elements. This thesis still uses a more generic effectiveness measurement by comparing the KPIs and the actual performance.

As identified during the evaluation session, the first proposed artifact (plateau aggregation metamodel specification) could lead into ternary relationship interpretation. The ternary relationship is something to be avoided, according to the Ontology/Semantic expert, due to its cardinality complexity. It creates confusion from the semantic perspectives because there could be three components being dependent on each other. Ternary relationship is something that is not aimed by the proposed artifact. Therefore, it is suggested to consider having relationship as a class by itself in ArchiMate specification. Currently, the ArchiMate differentiates core elements, which are represented as concept constructs, and core relationship. Introducing relationship as a class and integrate it in the proposed metamodel of the artifact would make the specification clearer and avoid ternary relationship.

Another interesting direction for future research is to explore the area of work package identification and prioritization. This research results in a proposed approach for gaps portfolio valuation as part of the Gaps, Solutions and Dependencies Matrix, as initially identified problem. Phase F of TOGAF's ADM aims to identify and prioritize work packages as the projects need to be implemented to close the gaps and reach the desired (target) architecture. By taking the prioritized gaps produced from the approach, another research could then focus on how to identify work packages to close the gaps according to the portfolio.

References

- ABACUS (2013). Enterprise Portfolio Management with ABACUS 4.1. White Paper of Avolution. Retrieved from: www.avolution.com.au
- Ahmed, M. D., Sundaram, D. (2012). *Sustainability Modelling and Reporting: From Roadmap to Implementation*. Journal of Decision Support Systems. pp: 611-624
- Aier, S., Gleichauf, B., Saat, J., Winter, R. (2009). *Complexity Levels of Representing Dynamics in EA Planning*. In Albani, A., Barjis, J., Diets, J. L.G. Advances in Enterprise Engineering III. pp: 55-69.
- Azevedo, C. L. B., Iacob, M. –E., Almeida, J. P. A., van Sinderen, M., Pires, L. F., Guizzardi, G. (2013). *An Ontology-Based Well-Founded Proposal for Modeling Resources and Capabilities in ArchiMate*. Enterprise Distributed Object Computing Conference (EDOC), 2013, IEEE 17th International.
- Bjekovic, M., Band, I., Kroese, R., Else, S. (2014). *ArchiMetal Case Study*. The Open Group.
- Buckl, S., Ernst, A. M., Matthes, F., Schweda, C., M. (2009). *An Information Model Capturing the Managed Evolution of Application Landscapes*. In the 21st International Conference on Advanced Information Systems (CAiSE09), Amsterdam. The Netherlands.
- Buckl, S., Ernst, A. M., Matthes, F., Schweda, C., M. (2008). *An Information Model for Landscape Management - Discussing Temporality Aspects*. In Johnson, P., Schelp, J., Aier, S., editors, Proceedings of the 3rd International Workshop on Trends in Enterprise Architecture Research 2008 (TEAR 2008), Sydney, Australia.
- Buckl, S., Ernst, A. M., Matthes, F., Schweda, C., M. (2009). *Visual Roadmaps for Managed Enterprise Architecture Evolution*. 10th ACIS International Conference on Software Engineering, Artificial Intelligences, Networking and Parallel/Distributed Computing.
- Buschle, M., Quartel, D. (2011). *Extending the Method of Bedell for Enterprise Architecture Valuation*. Enterprise Distributed Object Computing Conference (EDOC), 2011, IEEE 15th International.
- DeLone, W. H., & McLean, E. R. (2003). The DeLone and McLean Model of Information Systems Success: A Ten-Year Update. Journal of management information systems, 19(4), 9–30.
- Diefenthaler, P. (2013). *Interactive Roadmap Generation in Enterprise Architecture Planning*.
- Foorhuis, R., Steenbergen, M., Mushkudiani, N., Brusl, W., Brinkkemper, S., Bos, R. (2010). *On Course, but not There yet: Enterprise Architecture Conformance and Benefits in Systems Development*. Thirty First International Conference on Information Systems. Retrieved from: http://aisel.aisnet.org/icis2010_submissions/110/
- Gall, M. D., Gall, J. P., & Borg, W. R. (2003). Educational research: An introduction (7th ed.). Boston, MA: A & B Publications.

- Gartner. (2013). *Magic Quadrant for Enterprise Architecture Tools*. Retrieved from <http://www.gartner.com/technology/reprints.do?id=1-1MXD4M1&ct=131113&st=sb>
- Guizzardi, G. (2005). *Ontological Foundations for Structural Conceptual Models*. University of Twente.
- Guizzardi, G., Falbo, R., Guizzardi, R. S. S. (2008). *Grounding Software Domain Ontologies in the Unified Foundational Ontology (UFO): The Case of the Ode Software Process Ontology*. In Proceedings of the XI Iberoamerican Workshop on Requirements Engineering and Software Environments, pp. 244-251.
- Hilliard, R. (2000). IEEE-std-1471-2000 recommended practice for architectural description of software intensive systems. *IEEE*. Retrieved from <http://standards.ieee.org>.
- Hofer, S. (2013). *Modeling the Transformation of Application Landscapes*. In Grabis, J. et al. (Eds.): PoEM 2013, LNBIP 165, pp. 101–113.
- Hauder, M., Roth, S., Schulz, C., Matthes, F. (2013a). *Current Tool Support for Metrics in Enterprise Architecture Management*. Software Metrik Kongress.
- Hauder, M., Fiedler, M., Matthes, F., Wust, B. (2013b). *Analyzing Task and Technology Characteristics for Enterprise Architecture Management Tool Support*. 2013 17th IEEE International Enterprise Distributed Object Computing Conference Workshop. DOI 10.1109?EDOCW.2013.3, pp. 267-274.
- Iacob, M. -E., Franken, H., & van den Berg, H. (2007). *Enterprise Architecture Handbook*, Bizzdesign academy publishers.
- Iacob, M. -E., Jonkers, H., Quartel, D., Franken, H., & van den Berg, H. (2012). *Delivering Enterprise Architecture with TOGAF® and ArchiMate®*. Enschede: BiZZdesign.
- Iacob, M. -E., Jonkers, H., Quartel, D. (2012a). *Capturing Business Strategy and Value in Enterprise Architecture to Support Portfolio Valuation*. Enterprise Distributed Object Computing Conference (EDOC), 2012, IEEE 16th International.
- Johnson, P., Ekstedt, M., Silva, E., & Plazaola, L. (2004). Using enterprise architecture for cio decision making: on the importance of theory. *In proceedings of the Second Annual Conference on Systems Engineering Research*.
- Jonkers, H., Lankhorst, M. M., ter Doest, H. W. L., Arbab, F., Bosma, H., & Wieringa, R. J. (2006). Enterprise architecture: Management tool and blueprint for the organisation. *Information Systems Frontiers*, 8(2), 63–66. doi:10.1007/s10796-006-7970-2
- Jonkers, H., Quartel, D., & Blom, R. (2012). Business Models: Aligning Business Strategy and Enterprise Architecture [White paper]. *BiZZdesign*. Retrieved March 08, 2013, from: <http://www.bizzdesign.com/downloadmanager/download/4>
- Jonkers, H., Band, I., Quartel, D. (2012a). *ArchiSurance Case Study*. The Open Group.

- Land, M., Proper, E., Waage, M., Cloo, J., Steghuis, C. (2009). *Enterprise Architecture*. Springer.
- Lankhorst, M. (2009). *Enterprise Architecture at Work: Modelling, Communication and Analysis* (2nd ed.). Springer.
- Lapkin, A. (2007). What the C-suite Needs to Know about EA, and What the EA Team Needs to Know about the C-suite (Report no. G00167784). Retrieved from:
http://www.gartner.com/DisplayDocument?g_search&id=965412&subref=simplesearch
- Minoli, D. (2008). *Enterprise Architecture A Thru Z: Frameworks, Business Process Modeling, SOA, and Infrastructure Technology*. New York: Auerbach.
- Owen, M (2013). *Pragmatic Roadmapping with IBM Rational System Architect® and ArchiMate®*. Retrieved from:
http://www.corso3.com/files/6713/6188/1008/Corso_-_Roadmapping_White_Paper_1.00.pdf
- Peffer, K., Tuunanen, T., Rothenberger, M. a., & Chatterjee, S. (2007). A Design Science Research Methodology for Information System Research. *Journal of Management Information Systems*, 24(3), 45-77. doi:10.2753/MIS0742-1222240302.
- Quartel, D., Steen, M. W. A., Lankhorst, M. (2010). *IT Portfolio Valuation: Using Enterprise Architecture and Business Requirements Modeling*. Enterprise Distributed Object Computing Conference (EDOC), 2010, IEEE 14th International.
- Rose, W., B. (2005). Enterprises as Systems: Essential Challenges and Approaches to Transformation. *Systems Engineering*, 8(2), 138-150.
- Ross, J. W., Weill, P., & Robertson, D. (2006). *Enterprise Architecture as Strategy: Creating a Foundation for Business Execution*, Harvard Business School Press.
- Schekkerman, J. (2008). *Enterprise Architecture Good Practices Guide. How to Manage the Enterprise Architecture Practice*. Trafford Publishing.
- Schekkerman, J. (2011). *Enterprise Architecture Tool Selection Guide*. Institute for Enterprise Architecture Developments.
- Schuurman, P., Berghout, E. W., Powell, P. (2008). *Calculating the Importance of Information Systems: The Method of Bedell Revisited*. CITER WP/010/PSEBPP, University of Groningen. Sprouts Working Papers on Information Systems, <http://sprouts.aisnet.org/8-37>
- The Open Group. (2011). TOGAF® Version 9.1. Van Haren Publishing. Retrieved from pubs.opengroup.org/architecture/togaf9-doc/arch/
- The Open Group. (2012). ArchiMate® 2.0 Specification. Van Haren Publishing. Retrieved from pubs.opengroup.org/architecture/archimate2-doc/
- Turner, D. W. (2010). *Qualitative Interview Design: A Practical Guide for Novice Investigators*. The Weekly Qualitative Report, 3(15), -13.

- Urbaczewski, L., Mrdalj, S. (2006). *A Comparison of Enterprise Architecture Frameworks*. Issues in Information Systems. 8(2), 18-23.
- Venkatraman, N. (1997). *Beyond Outsourcing: Managing IT Resources as Value Center*. MIT Sloan Management Review, 38(3): 51-64
- Yin, R. K. (1993). *Applications of Case Study Research*. Applied Social Research Methods Series. Vol. 34. California: SAGE Publications, Inc. 131.

APPENDICES

Appendix A: The TOGAF Architecture Skills Framework

Table 19: Proficiency Level

Level	Achievement	Description
1	Background	Not a required skill, though should be able to define and manage skill if required.
2	Awareness	Understands the background, issues, and implications sufficiently to be able to understand how to proceed further and advise client accordingly.
3	Knowledge	Detailed knowledge of subject area and capable of providing professional advice and guidance. Ability to integrate capability into architecture design.
4	Expert	Extensive and substantial practical experience and applied knowledge on the subject.

Table 20: Generic Skills

Generic Skills	Roles								
	Architecture Board Member	Architecture Sponsor	Enterprise Architecture Manager	Enterprise Architecture Technology	Enterprise Architecture Data	Enterprise Architecture Applications	Enterprise Architecture Business	Program/Project Manager	IT Designer
Leadership	4	4	4	3	3	3	3	4	1
Teamwork	3	3	4	4	4	4	4	4	2
Inter-personal	4	4	4	4	4	4	4	4	2
Oral Communications	3	3	4	4	4	4	4	4	2
Written Communications	3	3	4	4	4	4	4	3	3
Logical Analysis	2	2	4	4	4	4	4	3	3
Stakeholder Management	4	3	4	3	3	3	3	4	2
Risk Management	3	3	4	3	3	3	3	4	1

Table 21: Business Skills and Methods

Business Skills & Methods	Roles								
	Architecture Board Member	Architecture Sponsor	Enterprise Architecture Manager	Enterprise Architecture Technology	Enterprise Architecture Data	Enterprise Architecture Applications	Enterprise Architecture Business	Program/Project Manager	IT Designer
Business Case	3	4	4	4	4	4	4	4	2
Business Scenario	2	3	4	4	4	4	4	3	2
Organization	3	3	4	3	3	3	4	3	2
Business Process	3	3	4	4	4	4	4	3	2
Strategic Planning	2	3	3	3	3	3	4	3	1
Budget Management	3	3	3	3	3	3	3	4	
Visioning	3	3	4	3	3	3	4	3	2
Business Metrics	3	4	4	4	4	4	4	4	
Business Culture	4	4	4	3	3	3	3	3	1
Legacy Investment	4	4	3	2	2	2	2	3	2
Business Functions	3	3	3	3	4	4	4	3	2

Table 22: Enterprise Architecture Skills

Enterprise Architecture Skills	Roles								
	Architecture Board Member	Architecture Sponsor	Enterprise Architecture Manager	Enterprise Architecture Technology	Enterprise Architecture Data	Enterprise Architecture Applications	Enterprise Architecture Business	Program/Project Manager	IT Designer
Business Modeling	2	2	4	3	3	4	4	2	2
Business Process Design	1	1	4	3	3	4	4	2	2
Role Design	2	2	4	3	3	4	4	2	2
Organization Design	2	2	4	3	3	4	4	2	2
Data Design	1	1	3	3	4	3	3	2	3
Application Design	1	1	3	3	3	4	3	2	3
Systems Integration	1	1	4	4	3	3	3	2	2
IT Industry Standards	1	1	4	4	4	4	3	2	3
Services Design	2	2	4	4	3	4	3	2	2
Architecture Principles Design	2	2	4	4	4	4	4	2	2
Architecture Views & Viewpoints Design	2	2	4	4	4	4	4	2	2
Building Block Design	1	1	4	4	4	4	4	2	3
Solutions Modeling	1	1	4	4	4	4	4	2	3
Benefits Analysis	2	2	4	4	4	4	4	4	2
Business Interworking	3	3	4	3	3	4	4	3	1
Systems Behavior	1	1	4	4	4	4	3	3	2
Project Management	1	1	3	3	3	3	3	4	2

Table 23: Program or Project Management Skills

Program or Project Management Skills	Roles								
	Architecture Board Member	Architecture Sponsor	Enterprise Architecture Manager	Enterprise Architecture Technology	Enterprise Architecture Data	Enterprise Architecture Applications	Enterprise Architecture Business	Program/Project Manager	IT Designer
Program Management	1	2	3	3	3	3	3	4	2
Project Management	1	2	3	3	3	3	3	4	2
Managing Business Change	3	3	4	3	3	3	4	4	2
Change Management	3	3	4	3	3	3	4	3	2
Value Management	4	4	4	3	3	3	4	3	2

Table 24: IT General Management Skills

IT General Management Skills	Roles								
	Architecture Board Member	Architecture Sponsor	Enterprise Architecture Manager	Enterprise Architecture Technology	Enterprise Architecture Data	Enterprise Architecture Applications	Enterprise Architecture Business	Program/Project Manager	IT Designer
IT Application Development Methodologies & Tools	2	2	3	4	4	4	2	3	3
Programming Languages	1	1	3	4	4	4	3	2	3
Brokering Applications	1	1	3	3	4	4	3	2	3
Information Consumer Applications	1	1	3	3	4	4	3	2	3
Information Provider Applications	1	1	3	3	4	4	3	2	3
Storage Management	1	1	3	4	4	2	2	2	3
Networks	1	1	3	4	3	2	2	2	3
Web-based Services	1	1	3	3	4	4	2	2	3
IT Infrastructure	1	1	3	4	3	2	2	2	3
Asset Management	1	1	4	4	3	3	3	2	3
Service Level Agreements	1	1	4	4	3	4	3	2	3
Systems	1	1	3	4	3	3	2	2	3
COTS	1	1	3	4	3	4	2	2	3
Enterprise Continuums	1	1	4	4	4	4	4	2	3
Migration Planning	1	1	4	3	4	3	3	2	3
Management Utilities	1	1	3	2	4	4	2	2	3
Infrastructure	1	1	3	4	3	4	2	2	3

Table 25: Technical IT Skills

Technical IT Skills	Roles								
	Architecture Board Member	Architecture Sponsor	Enterprise Architecture Manager	Enterprise Architecture Technology	Enterprise Architecture Data	Enterprise Architecture Applications	Enterprise Architecture Business	Program/Project Manager	IT Designer
Software Engineering	1	1	3	3	4	4	3	2	3
Security	1	1	3	4	3	4	3	2	3
Systems & Network Management	1	1	3	4	3	3	3	2	3
Transaction Processing	1	1	3	4	3	4	3	2	3
Location & Directory	1	1	3	4	4	3	3	2	3
User Interface	1	1	3	4	4	4	3	2	3
International Operations	1	1	3	4	3	3	2	2	2
Data Interchange	1	1	3	4	4	3	2	2	3
Data Management	1	1	3	4	4	3	2	2	3
Graphics & Image	1	1	3	4	3	3	2	2	3
Operating System Services	1	1	3	4	3	3	2	2	3
Network Services	1	1	3	4	3	3	2	2	3
Communication Infrastructure	1	1	3	4	3	3	2	2	3

Table 26: Legal Environment

Legal Environment	Roles								
	Architecture Board Member	Architecture Sponsor	Enterprise Architecture Manager	Enterprise Architecture Technology	Enterprise Architecture Data	Enterprise Architecture Applications	Enterprise Architecture Business	Program/Project Manager	IT Designer
Contract Law	2	2	2	2	2	2	2	3	1
Data Protection Law	3	3	4	3	3	3	3	2	2
Procurement Law	3	2	2	2	2	2	2	4	1
Fraud	3	3	3	3	3	3	3	3	1
Commercial Law	3	3	2	2	2	2	3	3	1

Appendix B: Architecture Development Methods (ADM) Overview

The TOGAF ADM describes a method for developing and managing the lifecycle of an enterprise architecture, and forms the core of TOGAF. In order to meet the business and IT needs of an organization, ADM integrates the elements of TOGAF that are described in the TOGAF publication as well as other available architectural assets, such as architecture descriptions, models, and patterns.

The ADM is iterative and thus, developing architecture is a continuous, cyclical process. The architect, while executing the ADM repeatedly over time, gradually adds more and more content to the organization's Architecture Repository. Even though the main focus of the ADM is the development of the enterprise-specific architecture, it can also be viewed as the process for populating the enterprise's own Architecture Repository with relevant reusable generic building blocks.

The first execution of ADM will be the hardest because the architecture assets that are available for re-use will be relatively scarce. However, architecture assets are available from external sources such as TOGAF and IT industry in general that could be leveraged in support of the effort. The following execution would then be easier because more architecture assets are identified and available for future re-use.

The structure of ADM has been previously shown in section 2.3.1. The phases of the ADM cycle are further divided into steps. The Requirements Management phases is a continuous phase that ensures all changes in the requirements are handled appropriately through governance process and reflected in all other phases.

Table 27: ADM Phases and Steps

ADM Phase	Steps
Preliminary Phase <ul style="list-style-type: none">Determine the Architecture Capability desired by the organizationEstablish the Architecture Capability	<ol style="list-style-type: none">1. Scope the Enterprise Organizations Impacted2. Confirm Governance and Support Frameworks3. Define and Establish Enterprise Architecture Team and Organization4. Identify and Establish Architecture Principles5. Tailor TOGAF and, if any, Other Selected Architecture Framework(s)6. Implement Architecture Tools
Phase A: Architecture Vision <ul style="list-style-type: none">Develop a high-level aspirational vision of the capabilities and business value to be delivered as a result of the proposed enterprise architectureObtain approval for a Statement of Architecture Work	<ol style="list-style-type: none">1. Establish the Architecture Project2. Identify Stakeholders, Concerns, and Business Requirements3. Confirm and Elaborate Business Goals, Business Drivers, and Constraints4. Evaluate Business Capabilities

that defines a program of works to develop and deploy the architecture outlined in the Architecture Vision	<ol style="list-style-type: none"> 5. Assess Readiness for Business Transformation 6. Define Scope 7. Confirm and Elaborate Architecture Principles, including Business Principles 8. Develop Architecture Vision 9. Define the Target Architecture Value Propositions and KPIs 10. Identify the Business Transformation Risks and Mitigation Activities 11. Develop Statement of Architecture Work; Secure Approval
Phase B: Business Architecture <ul style="list-style-type: none"> • Develop the Target Business Architecture that describes how the enterprise needs to operate to achieve the business goals, and respond to the strategic drivers set out in the Architecture Vision, in a way that addresses the Request for Architecture Work and stakeholder concerns • Identify candidate Architecture Roadmap components based upon gaps between the Baseline and Target Business Architectures 	<ol style="list-style-type: none"> 1. Select Reference Models, Viewpoints, and Tools 2. Develop Baseline Business Architecture Description 3. Develop Target Business Architecture Description 4. Perform Gap Analysis 5. Define Candidate Roadmap Components 6. Resolve Impacts Across the Architecture Landscape 7. Conduct Formal Stakeholder Review 8. Finalize the Business Architecture 9. Create Architecture Definition Document
Phase C: Information Systems Architectures <ul style="list-style-type: none"> • Develop the Target Information Systems (Data and Application) Architecture, describing how the enterprise's Information Systems Architecture will enable the Business Architecture and the Architecture Vision, in a way that addresses the Request for Architecture Work and stakeholder concerns • Identify candidate Architecture Roadmap components based upon gaps between the Baseline and Target Information Systems (Data and Application) Architectures 	<ol style="list-style-type: none"> 1. Select Reference Models, Viewpoints, and Tools 2. Develop Baseline Data Architecture Description 3. Develop Target Data Architecture Description 4. Perform Gap Analysis 5. Define Candidate Roadmap Components 6. Resolve Impacts Across the Architecture Landscape 7. Conduct Formal Stakeholder Review 8. Finalize the Data Architecture 9. Create Architecture Definition Document
Phase D: Technology Architecture <ul style="list-style-type: none"> • Develop the Target Technology Architecture that enables the logical and physical application and data components and the Architecture Vision, addressing the Request for Architecture Work and 	<ol style="list-style-type: none"> 1. Select Reference Models, Viewpoints, and Tools 2. Develop Baseline Technology Architecture Description 3. Develop Target Technology Architecture Description

<p>stakeholder concerns</p> <ul style="list-style-type: none"> Identify candidate Architecture Roadmap components based upon gaps between the Baseline and Target Technology Architectures 	<ol style="list-style-type: none"> Perform Gap Analysis Define Candidate Roadmap Components Resolve Impacts Across the Architecture Landscape Conduct Formal Stakeholder Review Finalize the Technology Architecture Create Architecture Definition Document
<p>Phase E: Opportunities & Solutions</p> <ul style="list-style-type: none"> Generate the initial complete version of the Architecture Roadmap, based upon the gap analysis and candidate Architecture Roadmap components from Phases B, C, and D Determine whether an incremental approach is required, and if so identify Transition Architectures that will deliver continuous business value 	<ol style="list-style-type: none"> Determine/Confirm Key Corporate Change Attributes Determine Business Constraints for Implementation Review and Consolidate Gap Analysis Results from Phases B to D Review Consolidated Requirements Across Related Business Functions Consolidate and Reconcile Interoperability Requirements Refine and Validate Dependencies Confirm Readiness and Risk for Business Transformation Formulate Implementation and Migration Strategy Identify and Group Major Work Packages Identify Transition Architectures Create the Architecture Roadmap & Implementation and Migration Plan
<p>Phase F: Migration Planning</p> <ul style="list-style-type: none"> Finalize the Architecture Roadmap and the supporting Implementation and Migration Plan Ensure that the Implementation and Migration Plan is coordinated with the enterprise's approach to managing and implementing change in the enterprise's overall change portfolio Ensure that the business value and cost of work packages and Transition Architectures is understood by key stakeholders 	<ol style="list-style-type: none"> Confirm Management Framework Interactions for the Implementation and Migration Plan Assign a Business Value to Each Work Package Estimate Resource Requirements, Project Timings, and Availability/Delivery Vehicle Prioritize the Migration Projects through the Conduct of a Cost/Benefit Assessment and Risk Validation Confirm Architecture Roadmap and Update Architecture Definition Document Generate the Implementation and Migration Plan Complete the Architecture Development Cycle and Document Lessons Learned
<p>Phase G: Implementation Governance</p> <ul style="list-style-type: none"> Ensure conformance with the Target Architecture by implementation projects 	<ol style="list-style-type: none"> Confirm Scope and Priorities for Deployment with Development Management Identify Deployment Resources and Skills Guide Development of Solutions Deployment

<ul style="list-style-type: none"> • Perform appropriate Architecture Governance functions for the solution and any implementation-driven architecture Change Requests 	<ol style="list-style-type: none"> 4. Perform Enterprise Architecture Compliance Reviews 5. Implement Business and IT Operations 6. Perform Post-Implementation Review and Close the Implementation
<p>Phase H: Architecture Change Management</p> <ul style="list-style-type: none"> • Ensure that the architecture lifecycle is maintained • Ensure that the Architecture Governance Framework is executed • Ensure that the enterprise Architecture Capability meets current requirements 	<ol style="list-style-type: none"> 1. Establish Value Realization Process 2. Deploy Monitoring Tools 3. Manage Risks 4. Provide Analysis for Architecture Change Management 5. Develop Change Requirements to Meet Performance Targets 6. Manage Governance Process 7. Activate the Process to Implement Change
<p>Requirements Management</p> <ul style="list-style-type: none"> • Ensure that the Requirements Management process is sustained and operates for all relevant ADM phases • Manage architecture requirements identified during any execution of the ADM cycle or a phase • Ensure that relevant architecture requirements are available for use by each phase as the phase is executed 	<ol style="list-style-type: none"> 1. Identify/document requirements - use business scenarios, or an analogous technique 2. Baseline requirements: <ol style="list-style-type: none"> a. Determine priorities arising from current phase of ADM b. Confirm stakeholder buy-in to resultant priorities c. Record requirements priorities and place in Requirements Repository 3. Monitor baseline requirements 4. Identify changed requirements: 5. Identify changed requirements and record priorities 6. Implement requirements arising from Phase H 7. Update the Requirements Repository with information relating to the changes requested, including stakeholder views affected 8. Implement change in the current phase 9. Assess and revise gap analysis for past phases

Appendix C: ArchiMate - Implementation and Migration Extension

Extension Metamodel.

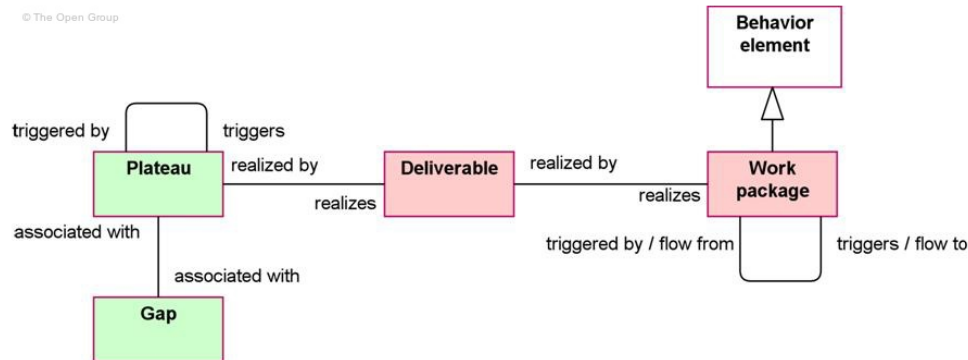


Figure 87: Implementation and Migration Extension Metamodel

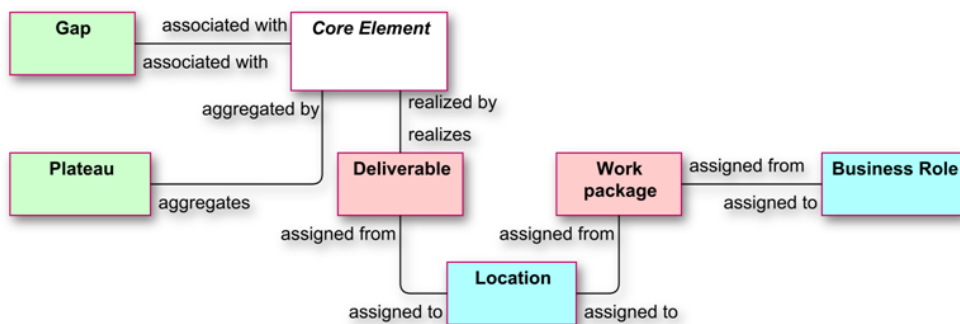


Figure 88: Relationship between Implementation and Migration Extension and the ArchiMate Core Concepts

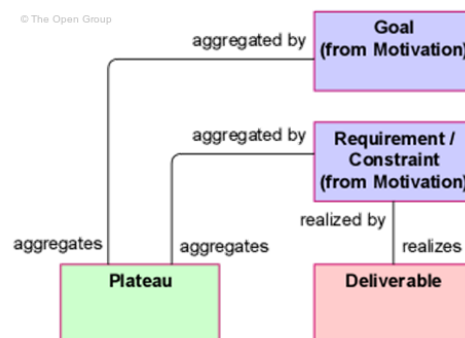


Figure 89: Relationship between Plateau, Deliverable and Motivation Concepts

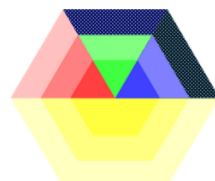
Extension Concepts.

1. **Work Package.** It is defined as a series of actions designed to accomplish a unique goal within a specified time. It has a clearly defined beginning and end date, and a well-defined set of goals or results. It can be used to model projects, but also sub-projects or tasks within a project, programs, or project portfolios.
2. **Deliverable.** It is defined as a precisely-defined outcome of a work package. It may be result of any forms, such as reports, papers, services, software, physical products, or intangible results like organizational change. It might also be the implementation of (a part of) an architecture.
3. **Plateau.** IT is defined as a relatively stable state of the architecture that exists during a limited period of time. It is used to represent the various states of architectures such as Baseline Architecture, Transition Architecture and Target Architecture.
4. **Gap.** It is defined as an outcome of a gap analysis between two plateaus. It forms an important input for the subsequent implementation and migration planning. Gap concept links the two plateaus and shows the differences between them.

Extension Viewpoints.

1. **Project viewpoint.** This viewpoint is mainly used to model the architecture change management. The migration process has significant consequences on the medium and long-term growth strategy and the subsequent decision-making process. Some of the issues that should be taken into account by the models designed in this viewpoint are:
 - a. Developing fully-fledged organization-wide enterprise architecture is a task that may require several years.
 - b. All systems and services must remain operational regardless all the presumable modifications and changes of the enterprise architecture during the change process.
 - c. The change process may have to deal with immature technology standards, for example messaging, security, data, etc.
 - d. The change has serious consequences for the personnel, the culture, the way of working and the organization.

Table 28: Description of Project Viewpoint

Stakeholders	(operational) managers, enterprise and ICT architects, employees, shareholders	
Concerns	Architecture vision and policies, motivation	
Purpose	Deciding, informing	
Abstraction Level	Overview	
Layers/Extensions	Implementation and Migration extension	
Aspects	Passive structure, behavior, active structure	

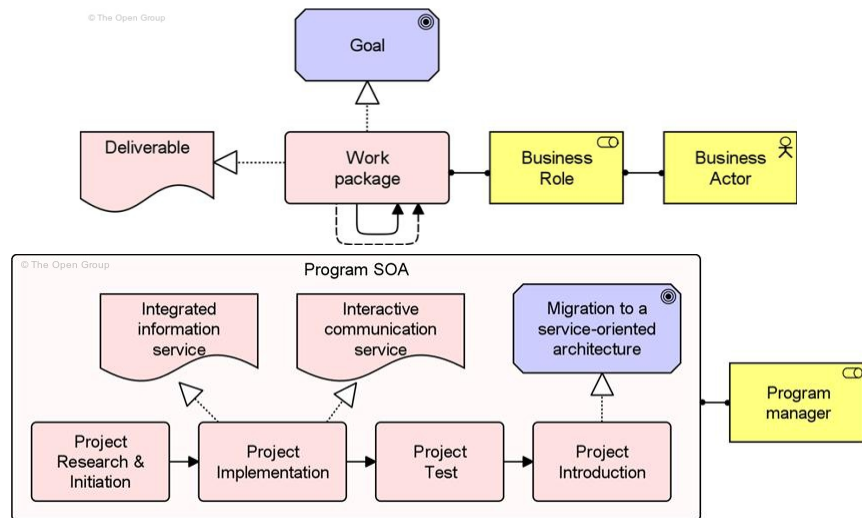



Figure 90: Concepts and Relationships of Project Viewpoints

2. Migration viewpoint. It is used to model the transition from an existing architecture to a target architecture. Thus, it includes models and concepts that can be used for specifying the transition process.

Table 29: Description of Migration Viewpoint

Stakeholders	Enterprise architects, process architects, application architects, infrastructure architects and domain architects, employees, shareholders	
Concerns	History of models	
Purpose	Designing, deciding, informing	
Abstraction Level	Overview	
Layers/Extensions	Implementation and Migration extension	
Aspects	Not applicable.	

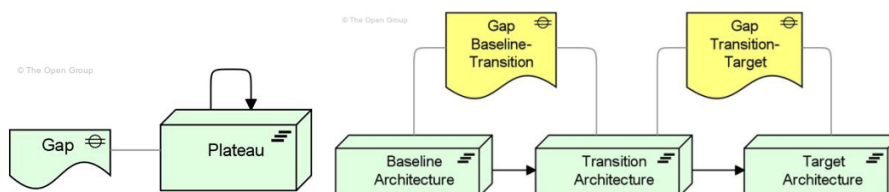
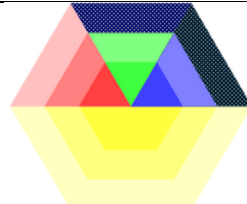


Figure 91: Concepts and Relationships of Migration Viewpoints

3. Implementation and migration viewpoint. It is used to model the relationship between the programs and projects and the parts of the architecture that they implement. This means that it allows modeling of the scope of programs, projects, project activities in terms of plateaus that are realized or the individual architecture elements that are affected. This viewpoint can be used in combination with the programs and projects viewpoint to support portfolio management:
 - a. The programs and projects viewpoint is suited to relate business goals to programs and projects. For example, this makes it possible to analyze at a high level whether all business goals are covered sufficiently by the current portfolios.
 - b. The implementation and migration viewpoint is suited to relate business goals (and requirements) via programs and projects to (parts of) the architecture. For example, this makes it possible to analyze potential overlap between project activities or to analyze the consistency between project dependencies and dependencies among plateaus or architecture elements.

Table 30: Description of Implementation and Migration Viewpoint

Stakeholders	(operational) managers, enterprise and ICT architects, employees, shareholders	
Concerns	Architecture vision and policies, motivation	
Purpose	Deciding, informing	
Abstraction Level	Overview	
Layers/Extensions	Business layer, application layer, technology layer, implementation & migration extension	
Aspects	Passive structure, behavior, active structure	

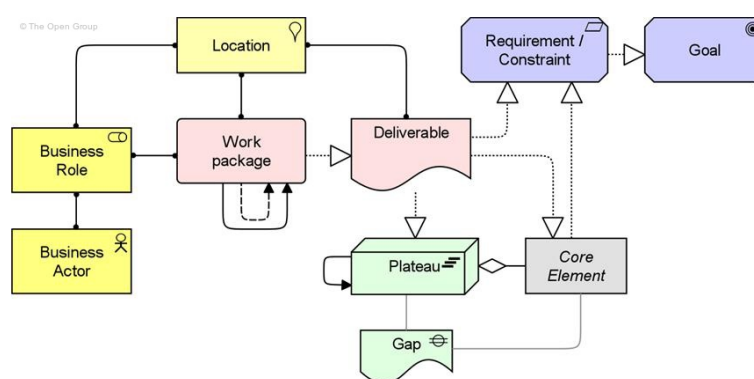


Figure 92: Concepts and Relationships of Implementation & Migration Viewpoints

Appendix D: Interview Transcripts

The following transcripts are written as the text products of the interview sessions that have been conducted with the three respondents. The interview sessions were conducted as the means of evaluation purpose as described comprehensively in Chapter 6.

Interview 1

Interviewee's function/role:	Ontology/Semantic Expert (Interviewee 1)
Interview date:	Wednesday, July 2, 2014 (16.00 – 17.00)

Artifact 1: Adding relation aggregation to plateau concept

From the theoretical (formal) perspective, does the solution provide sufficient level of correctness? What needs to be improved?

You would have to map it. You would have to define that relation as a class itself, to draw this better. This relationship would rather be a class, like a (concept) model. The way it is drawn now is like a kind of ternary relations, and that is not what you want. You want this relation to be independent of these other relations and then you want to have this relation (to be drawn). So, you have to define the relation itself. In ArchiMate, the way the (core) relationships are defined is different from the core concepts. And there is a challenge there that you have to find a fit to apply the changing, somehow. I understand the objective of the solution and in terms of the metamodel technique, I would represent them as an addition tree of relations: a relation at the top and now the specialized relations at the bottom so then we can have a class. We talk about the class of relations: "A relation class". Just like you have core elements, now you have core relations as a class. It seems to me the correct way.

I think there is a metamodeling difficulty in using the ternary relations as such, because you create a dependency between the components. From looking at your solution, you do not want to say that there is ternary relationship between these three components (two applications and a plateau). That is not what you want. You want to say that there is this relation and there is THIS OTHER relation.

To me you have to start treating relations as first class sits on your domain of your quest. For me to do that, normally you have to create a class to represent it. Maybe I am collapsing some of the answers to the questions. This is a really interesting initiative to the ArchiMate.

I don't think there are many ternary relationships in ArchiMate specification. We really don't want ternary specification, not only it complicates the metamodeling. Many metamodeling techniques use binary specification because it is so much simpler to deal. When we have ternary specification between components, the cardinality constraints is a disaster and the interpretation of it would be very hard from the semantic point of view.

We don't want to couple them, we want to separate things and you want these two relations as two binary relations. We don't this to be interpreted as a relation between two core element and the plateau. We want two different relations: (1) between the core element and another core element, and (2) between the relation and the plateau.

This is an interesting topic because there is no metamodeling approach that does exactly THIS but unless you are talking about relations explicitly. The constraint or requirements about the source and target components when applying the solution is logical and makes sense.

Do you think it is feasible for the approach to be included in the ArchiMate specification?

It is not a minor kind of thing. You can always change the ArchiMate specification. The ArchiMate specification does not really start with a very clear metamodeling infrastructure. It is an ad-hoc thing. So it does not tell you whether this is prohibited or not. It is like when we are using ECore metamodeling. It does not tell what the language is in this to draw the metamodel, it is implicit.

So, if you show this to someone and ask, "Is this right?", I don't know, really because there are no rules. This is feasible to be implemented and require some changes because to talk about relationship that is explicitable, of course you can always do. There are many questions inside this question.

Relationship in ArchiMate is not yet represented as a class. It could be a bit intrusive but it seems like we need this to treat relationships as elements, some kind of constructs. Like we need super class construct on top of elements and relations.

The fact that BiZZdesign Architect already treats relationship as an element is very good for implementation. Now it is just the matter of adjusting the language (metamodel) to also have that. In this case the tool seems to be built from a better language engineering principle than the specification, which is to refine the association. I think it is a very good approach as we can now relate relations and can specialize relations and do things with it.

Do you think this solution is useful in practice? Why?

There could be other cases where the solution is applied, like capability extension. We want to say that all these elements together and are arranged in this way, they realize a capability. So, for instance, we need to put these elements with this relation (similar solution implementation). This could be shown by having a "grouping". But again the problem with the specification is that "grouping" is treated as a relation, not as a concept.

In my opinion, the relation between plateau and the core elements is not semantically the same as aggregation relationship. The aggregation relationship could represent many different meanings and it is overly misused in practice. When we have an application that is part of another application, it is different type of semantic meaning when we also say that an application is part of a plateau. The meaning of aggregation in these two "part-of" relationship is different but the symbol is the same. I prefer to see two different kinds of relations to represent these situations. Actually what you want to say is there is a new kind of relation between plateaus and core elements and core relations. All of them. This would be the idea.

To me it is a different semantic relation. It is no whole-part relation. Components are not part of a plateau, they exists or are present at a plateau. It is more precise way of saying because plateau is a time thing. Plateau is a state of an architecture and we want to talk about the plateau. That is why, it is made a construct. But, the relation between plateau and other core elements is semantically different from a whole-part relationship.

In general, I agree the idea to include the relations between relations but "aggregation" in plateau is special relation that says that some components and relations exist in plateau. It is different type of relation than aggregation. I would not say it as an aggregation. Because I am very much concerned with the semantic, I would prefer something more specific here, which is the (special) relationship between the plateau and the other elements. We introduce the concept of plateau, why not introducing a relation.

Maybe you could look at "named graph" in UML kind of stuffs. It is also tries to represent or take a framework of a graph and give that some properties and names and talk about that. It is like you zoom and select some part of graph and talk about this fragmented graph. It could be a nice resource to help redefine your solution.

Artifact 2: Gaps Portfolio Valuation approach

Plateau namings for the alternative paths should be modified to represent the real situation. "Integrated CRM system" and "Integrated Back Office system with integrated CRM system". This ("Integrated back office system") is not the same as this ("Integrated Back Office system with integrated CRM system") and not good to give them the same name.

Step 2

These goals are "change goals"? I am trying to understand your approach overall. So if you have these different states (baseline and target), so you have to identify your goals for doing this, for doing the change?

To me it is a bit confusing way to represent what you want to do. Because these goals (improve data consistency and reduce maintenance cost) are what motivate you to go from this (state 1) to that (state 2). They are what motivate the change, so not (yet) realized by the service.

Can you know if a component (that exists in both plateaus) is modified somehow? Having a "new" construct to represent versioning is bad.

Step 5

Effectiveness meaning?

Is it just a coincidence that the sum of the green ones (the weights of the realization relationship) is 1? What is the green one?

I (now) understand this effectiveness. You talked about how it currently matches the performance, but then many criteria can be used. ← showing the measurement types for each layer.

Find the representation of goals and the relations with the gaps confusing conceptually. Because we have these kind of change goals. You have some goals and you have architecture "as-is" and then you decide to change something because you have some "change goals" to go from where you are to where you want to be. And then you identify the gaps and, these "change goals" would be realized if you implement these change (closing the gaps).

And then when you say that goals are realized by the services and you do this grouping. To me, the grouping strategy is a bit artificial. What does it really represent? I am bit confused with that.

It is always the challenge how to validate the numbers. You have this number crunching and how do you know that these are at least something. This is just a general thing and always a problem for research. In the end people get some recommendations in order to do this (transformation path). Do you really know if that is the best one? To me, it is always a problem. Nice, nice stuffs!

Interview 2

Interviewee's function/role:	Enterprise Architect Expert (Interviewee 2)
Interview date:	Thursday, July 3, 2014 (14.00 – 15.00)

Artifact 1: Adding relation aggregation to plateau concept

From the theoretical (formal) perspective, does the solution provide sufficient level of correctness? What needs to be improved?

Yes, it is a problem that it is not aligned with the (ArchiMate) standards when we want to also aggregate a relationship to the plateau. Grouping is a type of relation in ArchiMate to show that several components share something in common but it is also not clear whether grouping includes relationship as well.

Aggregating a relationship (between components) to the plateau is something that does not exist yet in ArchiMate but we can do that in Architect. I agree with the conditions of both source and target components in aggregating the relationship between them to the plateau. They both have to be valid in the same plateau. Otherwise that does not make sense.

Do you think it is feasible for the approach to be included in the ArchiMate specification?

I think so. It is something that does not exist yet in ArchiMate. So, it is something new of course, a relation to a relation. And this could also be used for another situation. And of course you can associate this with nesting as another representation. Because it, the nesting, is also often used to represent aggregation. Yes, nesting could also represent composition beside aggregation. But in case of plateaus, there is no composition. So it can only be aggregation.

But I see this as an alternative to represent the same model. So, I think it is still correct. In the current ArchiMate metamodel specification, there is no definition yet that we can aggregate relation to relation but in practice we need this. So, there is a motivation to this situation. Yes, I think it is feasible to be applied. Architect tool can already do this. It is just only the ArchiMate that does not this additional relationship yet.

Do you think this solution is useful in practice? Why?

Yes. I think the solution can also be used in other situations. For example there are two applications having the flow relationship from one to the other. We want to know what kind of information that is exchanged here. You would need to associate this flow relationship to make it explicit what kind of information that has exchanged here. So I think this solution could also be useful in this situation.

Imagine if the information being exchanged is of data object or business object type. We still need to associate this to the flow relationship

Artifact 2: Gaps Portfolio Valuation approach

From the theoretical (formal) perspective, does the solution provide sufficient level of correctness? What needs to be improved?

The calculation is hard for me to judge. I am not that familiar with the Bedell's method but I understand the overall steps in general. Calculation is the difficult problem to do, especially when considering the target architecture and the performance of the components in the target architecture. Difficult to establish.

Do you think this method is useful in practice and give an insight to the user in managing their architectural gaps?? Why?

I think it is useful to the users. Some part of the steps could be made automated which made it is even more useful for the users for their decision making process.

Especially when all of the architecture of the enterprise has been well structured and well documented.

I think I mention everything during your presentation. I would like to see what you can do with the results. See how you go with the proposal for improvement change in ArchiMate later

Comments and feedback during open discussion:

Identifying the gaps itself (between baseline and target) is still part of phases B C and D.

Gap classification ← be careful not to mix it with specialization. Because these are just the names of the gaps and have no formal meanings. So you can't really specialize or classify them.

Goals – gap components relation ← There would be a case, I could imagine, that a component might contribute to more than one goals. So, in this case study each gap component relates to only one goal. In different (real life) cases, a component could realize more than one goal.

Step 3. Normalization ← is normalization a correct phrase/way to describe the step? Because normalization is normally also changing the structure according to standard set of structure model. . Maybe "filtering" is better to represent filtering out the elements that are not relevant.

Step 5. Effectiveness

← It does not have to be 1 in total. Maybe, I am not sure; this is not complete in all cases. I could imagine that it could be lower than 1 where not all relations are shown in this ("normalized") model. So it should be 1 in the original model and could be lower than 1 in the normalized model.

← A complete model should only be in 1 plateau. In the normalized model, it is better to separate the model based on baseline/target architecture. To see the distribution of the weight. A server could support different applications that exist in different states. The total of the weight should be calculated considering different states the components are in.

Interview 3

Interviewee's function/role:	Internal Consultant (Interviewee 3)
Interview date:	Friday, July 11, 2014 (13.30 – 15.00)

Artifact 1: Adding relation aggregation to plateau concept

From the theoretical (formal) perspective, does the solution provide sufficient level of correctness? What needs to be improved?

Yes, this is indeed a problem that there is no a direct line in the metamodel specification from the plateau to the relationship between components that belong to the plateau. The fact that we can actually assigned properties from relationship to the plateau is something we invented in our tooling but it is not what the standard proposes.

Is the solution (extending the metamodel spacification) something that is also familiar in other modeling languages, such as UML? Normally relationship is represented as a concept then it is easier to draw the metamodel. But in ArchiMate there is no clear distinction between concept construct and core relationships, relationships are not represented as a construct.

Taking grouping as a way to represent plateau concept is not solutive because it's just a box without meaning. Grouping is a type of relationship not a concept. Using grouping to represent plateau somehow skips the plateau concept. We aggregate the grouping to the plateau.

Of course, I think, it can be a solution to address the aggregating a relation problem. So, this is possible. I think it is a correct solution and it is needed. It has a high impact.

Do you think it is feasible for the approach to be included in the ArchiMate specification?

I am not sure if it is feasible to be included in ArchiMate specification because you change something to the basics of the language. Let's say we're gonna add simply add line to connect a relation to a relation and I think it's something possible but it's quite a change in the nature of the ArchiMate language. It is a fundamental change, I think. And that is quite an impact, in that sense.

But this is a correct solution, I think. This is needed and this has a high impact. Because I don't see it in the other (alternative) solutions as they are not where the ArchiMate want to go to. Our previous solution that we just add properties to concept AND relations. That's what we already had in our tooling before ArchiMate had this plateau concept invented. I think that is the only reasonable alternative. But also there we could have a problem because ArchiMate is not intended now to specify attributes to the concept. We (ArchiMate) don't specify attributes, do it yourself. So they would not be too happy to add attributes in its specifications but they will not probably too happy to add this (the artifact) but this is probably closer to the current specification than adding properties. I think.

This solution (artifact) could be the best solution since it is not only serving the customers of BiZZdesign Architect but all the users all over the world. That would be my preferred solution. So, this is the best way to go, because something like this would be added in the specification then it is for all ArchiMate users and all tool vendors who have implemented ArchiMate should then be able to implement this in their tool environment.

Do you think this solution is useful in practice? Why?

The same utilization could also be applied in a situation where there are two applications having a flow relationship between them. We can use this to detail out what kind of information being exchange. Now we do it with the properties on a relation and we link it that way to a data object. Data object specifies

information or data that is exchanged between applications. That could be another example of how you could use this way of modeling. This is a question I get quite often (from users/clients). I can also imagine that in general, this concept gives more opportunities to model complex stuffs.

It may be that it gives more option to model difficult or advanced issues. I think, it is then therefore useful in practice. On the other hand it makes the language more complex. That is always the trade-off. When the language is more complex, it gives you more opportunities to be more precise in what you want to specify. I don't see a problem in that. If you don't need it then you don't use it.

One thing that should be noted here is that if you really want to ensure if this is feasible to be included in ArchiMate specification is to go to the Open Group and you get the whole complex process of all the members of the ArchiMate forum. They want to discuss and they have opinions about certain initiatives. If they don't agree about it, then it could not be there. So, it is quite a long and complex process to really get the change or update in the ArchiMate specification and get it finalized. So that maybe a practical problem to get something like this implemented for real. Because how it works is that you have this ArchiMate forum where all kinds of organizations are members. Henry is the chair of the forum, so we have a bigger influence. However, all of the forum members have their influences.

You can consult Henk or Henry about this ArchiMate forum. This is more about how the ArchiMate forum works, how the standard is updated. I am not sure if this should be the scope of your thesis. I think it is good to have some thoughts about it in your recommendation.

Artifact 2: Gaps Portfolio Valuation approach

Normally, after phase B, C, and D are done, when the gaps are collected, there is no real standard approach to come up with work package. It depends first what is the scope, what is my baseline and what is my target, is it 1 year difference or 5 year difference. Work packages are collection of elements in your target architecture which basically could be grouped together to be implemented in a project. In your target architecture, you may have some new elements, some updated elements, and some removed elements, and you have to argue and think about what do we have then and how do we organize them. It makes sense that elements that are really depending on each other directly are taken into one work package, because they are more related and candidate to be part of the same work package.

The other way is to see from vertical way. For example, we have 10 products to deliver. And everything that is depending to deliver product A, either business process, or application or infrastructure, we are going to take that as one package. That is another approach. A bank that is going to offer an insurance product, then we introduce insurance work package. Everything that is part of the insurance is then now built or everything that is in insurance is now updated. I see there are two approaches: (1) a vertical, from business perspective, new product, new services, or (2) horizontal, things that are closely connected, things that are impossible not to stay together. I think those are two main aspects in identifying work package.

From the theoretical (formal) perspective, does the solution provide sufficient level of correctness? What needs to be improved?

I think we have already discussed a lot. In general, I think this kind of approaches indeed a good thing to calculate. But it is not everything. You always need to take into account all elements that can be reasons to do something first. Another one could also be, but I would not take that into the methodology, that in practice what you see often is that these decisions are not always taken on objective and neutral arguments but more on feelings. I don't say it is right but it is what often happens in practice.

So I think it is something you should mention that if you want to introduce this kind of methodology in organization, people will not like this because it will make everything very transparent. A lot of people don't want that because when they have power, they could base their decisions based on their feelings

then they are much happier. So, that is always a downside for this kind of method. I like this kind of approaches but there are people that don't like it. It is like a warning when implementing this kind of methodologies that some people won't like it. But that's OK. I think it would be good for companies that they would work towards this.

Based on your experience, does the method include all the required activities in practice?

I think this should always be a combination. So, I think for (Enterprise) Architect, who should do this work, it is really good stuff. They have another job to do: to think about how they are going to manage all stakeholders in this process. In the end it's not always only about showing them but also about explaining the tasks behind it.

The way I see it, this methodology is only for Architect for doing this work, and then there should be another step, which you don't have to work it out in detail, that is about how are we going to give and present this results to all kind of stakeholders in organization. That will not end up that we have just three slides with the numbers, but it is more important that you convince those people with a good story. Back up the numbers with a good story. Then you have a good presentation. Then it might convince those stakeholders.

But if you start with: OK, we have calculated and this is the results, so you have to go for A. Then it might not work. The difficult thing with numbers is that you try specifying why or how numbers are chosen. The more like those educated guesses that you do, the less valuable the results of course would be. People are starting about to debate that the numbers are not correct. This is also always a bit of a downside of this kind of process that people can discussion about numbers you used. But then again I say, you need to combine it with a story. The numbers are just the indications of what we look at, and this the story behind it: so a total picture. That's important.

Is the method clear? Which steps need further elaboration?

How do you calculate the importance of the gap components? It looks like what Lianne, R&D, is doing. She's working on the portfolio management stuffs. She is thinking about defining all kinds of matrices to give some kinds of business values to elements of your portfolio. There you see that for example in the application component, we can assign business value as a number in your model. This gives all kinds of mathematical analysis. Then you have to determine why the business value for application A is 6. Still, you need to talk to the users or think about which business products that this application supports. Maybe these matrices are also interesting for you (to determine the importance).

Step 2. Does this step always start from bottom-up, from technology layer to the upper domain? It depends on the gaps components. For example if there is a gap component in business domain, then we can trace it down to the application domain and infrastructure domain.

Step 4. Do you answer these questions (calculating the importance guideline) by just common sense? OR you could also probably use your architecture models to answer these questions. In this way, you need to specify clearly in your architecture model, which business process is, for example, strategically critical to the organization. Because the way you do it now is simply use the guideline as a way to debating about it (the values).

Step 5. I am not so familiar with this kind of working (effectiveness measurement types), but it seems so technical. I can imagine that we could apply this mathematics (measurements) towards application components or services. In other cases, when I talk about new business process or business actors, then I am not sure whether this will work. For example the "completion time" here, is this a time for a request that is asked to an application? Is this technical? Or can we treat it like a customer calls to the company? So, the main thing here is "are these types of measurements defined only in IT technical metrics, or can you see them also to be more business metrics?" Because the way I see it now, these measurements are more technical.

How do you determine the values of the weight of the arrow? In practice, you could make it more precise by using the real statistics from the operational database sources that clearly show how many transactions are there supported by specific application. From this operational data sources, we could then have more detailed values on the relations. Then you will need to think about which variables you are going to look at.

Step 6. Do you also look which relationship in taking into consideration for the interdependency level? ArchiMate has specified table of the relationships' strength. For example, an association is a very weak relation. You can associate almost everything with everything. In that sense, association relation is given lower value of strength. It does not mean that much if a component has so many association relationships. Used-by relationship is stronger, because if you are going to look at interdependency level, a component with only one used-by relationship could be still considered has higher interdependency level as compared to another application with five association relationships.

ArchiMate has only specified the weight for structural relationships. So, flow and trigger relationships are not part of that. You need to figure a way out of dealing with that. I think it is worth looking at that, especially with these associations because they do not say anything. The initial intention of the relationship weight table is to show derived relationship. But maybe this could be a new way of using this table.

Step 7. We consider the groups that are more complex will be given more concern and thus given more priority. That is a way to look at it. But I could also argue the other way around. Let's say I have two groups and they have the same score for importance – effectiveness gap. Group A is far more complex than Group B, then it has a higher score for the interdependency. Then as a business or decision maker, I would say let's start with Group B because it is less complex then less risk but we have the same post-implementation value. Start always with the easiest one.

In practice, they have exactly done it. The decision was to start with the easy one. Sometimes, this is just business decision because what all of the business managers and project managers talk about is RISK. We do not want too much risk. If we model something that we choose between two projects, given they both provides the same value, then I would always choose the one with less risk.

Starting with the more complex gaps is also good. Separating both concerns (importance-effectiveness gaps and interdependency) and not put them together n overall score could also be an option. Another criterion could also be the "obligatory" or regulation that we must implement something. You see, as a government defines a new law and this result to an implementation of system. This could be very complex and maybe not that much value, but we have no choice. We have to do it because it is a law. Another criterion could be size, how large would a project be. In principle, the larger a project, the more difficult it is.

I would always mention that this is good to have these values/score as you have now. But it is also good to mention other reasons from business and law is an example that would overrule these scores. And I would say if I have the simple one: that is the first thing I do.

Comments and feedback during open discussion:

(Gap concept utilization)

"Modified" gap is q tricky one because we create a new object (concept) to represent that an object is updated so it's a difficult thing there. Now they have the same name where practically, you now create a new object. Let's say I have a version number in the original one. Then I copy it and paste it as a unique copy. If I rename the new copy, then I don't know anymore that it is a new version of the previous one because it is a new entity. Of course if you have this relation (relation through the "modified" gap), you can always ask through this to do back trace analysis. Ask to this new component, where do you originate from. Then give us answer of the object that is the original one.

But indeed, it gives more insights to have these types of gap. This one would actually be quite helpful.

Maybe you should reconsider why aggregation is used to represent the gap classification. You can always argue that I have four types of gap, which would be “removed”, “added”, “replace” and “added”, and then it would be a SPECIALIZATION. The nice thing about that approach is that ArchiMate language allows you to create your own specialized concepts of existing concepts. So you are allowed to exchange of an exactly ArchiMate language for my own purpose. So the gap concept is already a defined concept then you are free to specialize this gap concept into these four types. It is one of the language extension mechanisms. This approach would not need an update in the ArchiMate language because you are already allowed to specialize ArchiMate concept for own purpose. I like this idea.